CLUBE

Dezembro/84

N.º 27

Coordenadores: MARIA IRENE • ALEXANDRE SOUSA • J. MAGALHÃES • AV. BOAVISTA, 832-2.º T. • TELEF. 65127 • 4100 PORTO

NESTE NÚMERO

INTRODUÇÃO À LINGUAGEM MÁQUINA (Cont.) 1
HARDWARE 3
ROTINICES 4

Programas Spectrum/New Brain

- Matemática/Raizes 8
- Definidor de Caracteres 8
- Jetset Willy 11
- Basic Expansion 12
- Colditz 13
- Achou/Ganhou 13
- Dados 14
- Desenho Técnico 15
- Mensagem em Movimento 16

SPECSOUND 17
DESAFIO 18
NOVOS PROGRAMAS 19
PERGUNTAS/OBSERVAÇÕES/COMENTÁRIOS 20
RS 232/INTERFACE PARA COMUNICAÇÕES 21
TOP CLUBE Z80 21

No interior:

Folheto Mercado Z80

Edição: Clube Z80

Fotocomposição: Fotomecânica Mabreu/Porto

Impressão: Ramos dos Santos & C.ª, Lda./Porto

Tiragem: 500 exemplares, Dezembro 1984

INTRODUÇÃO À LINGUAGEM MÁQUINA

ZX81/SPECTRUM

Autor: FERNANDO PRECES
SACAVÉM

(Cont. dos números anteriores)

PARTE III — COMO FUNCIONA O Z80

4.3. — As mnemónicas do z80 (Continuação)

3.º Subgrupo **As instruções DEC**

DEC A	61
DEC H	37
DEC L	45
DEC B	5
DEC C	13
DEC D	21
DEC E	29
DEC (HL)	53
DEC (IX + d)	221, 531 + d
DEC (IY + d)	253, 53 + d
DEC HL	43
DEC BC	11
DEC DE	27
DEC SP	59
DEC IX	221, 43
DEC IY	253, 43

Tal como as instruções INC (Grupo 6), as DEC são muito utilizadas quando se trabalha com contadores do tipo FOR-NEXT ou com registros apontadores. Elas afectam todos os flags, com excepção do carry. Serão fornecidos ao leitor detalhes sobre a movimentação desses flags quando abordarmos os contadores.
Os tempos de execução das instruções DEC são idênticos às INC (ver grupo 6).
As instruções DEC (HL), DEC (IX + d) e DEC (IY + d), subtraiem uma unidade (1) ao conteúdo do endereço memorizado por qualquer destes registros. As restantes subtraiem 1 ao número memorizado pelo respectivo registro.
A diferença entre estes dois tipos de instruções situa-se no uso do parêntesis.
Como existem instruções para decrementar individualmente cada uma das células dum registro par, executando a instrução DEC sobre o byte mais significativo, podemos obter saltos de 256 unidades para trás. (Ver ensaio 2). Poderemos ver a utilidade destes saltos, quer para a frente INC, quer para trás DEC, quando trabalharmos com blocos de dados de extensão inferior ou igual a 255 bytes.

Ensaio 1: Decremento de N (N-1)

LD BC, 256	1, 0, 1
DEC BC	11
RET	201

PRINT USR X (Resultado: 255)

Ensaio 2: Decremento de N (N — 256)

LD BC, 511	1, 1, 1
DEC B	5
RET	201

PRINT USR X (Resultado: 255)

Ensaio 3: Decremento do conteúdo dum endereço

LD HL, 28020	33, 116, 109
LD (HL), 200	54, 200
DEC (HL)	53
RET	

RANDOMIZE USR X
PRINT PEEK 28020 (Resultado: 199)

Grupo 8 — As instruções de comparação

As instruções deste grupo são frequentemente usadas na programação em C/M. Elas permitem comparar um dado + N, conteúdo dum registro, ou o conteúdo dum endereço da memória, com o valor fixado no **registro A.**
A comparação efectua uma operação de subtracção cujo resultado vai afectar o estado dos flags, **mas que não altera o conteúdo de A.**
Em certa medida este tipo de instrução compara-se ao comando Basic IF, pois ao aplicá-la é implícito o uso de outro tipo de instrução que obrigue o Z80 a uma opção. Esta é tomada (seja ela qual for) com base no estado dos flags **Carry** e **Zero.**

Para o resultado duma instrução de comparação:

a) igual a zero — Flag Z = 1
b) <> de zero — » Z = 0
c) >= a zero — » C = 0
d) < que zero — » C = 1

Mnemónicas	Códigos
CP + N	254, + N
CP A	191
CP H	188
CP L	189
CP B	184
CP C	185
CP D	186
CP E	187
CP (HL)	190
CP (IX + d)	221, 190, + d
CP (IY + d)	253, 190, + d

Falarmos sobre o tempo de execução destas instruções não tem significado lógico, pois não existe alternativa para elas.

Para compararmos o conteúdo do registro A com qualquer outro valor, este é o processo mais rápido e eficiente. Existe ainda outro tipo de instruções de comparação (executado em bloco), que serão examinadas noutro grupo.

GRUPO 9 — As instruções lógicas

Em C/M existem 3 tipos fundamentais de instruções lógicas (AND, OR, XOR). Elas são executadas entre o conteúdo do registro A e um byte específico, tal como, um comando directo (byte seguinte à instrução lógica, denominado + N), um valor fixado num registro ou ainda o conteúdo dum endereço de memória apontado por um registro. Estas instruções obrigam o Z80 a operações bit a bit, sendo o resultado dessas 8 operações distintas devolvido ao registro A.

Subgrupo A — Instruções **AND**

A operação AND determina que o bit resultante do teste entre 2 bits, seja 1 (Set) apenas quando ambos tiveram o valor 1. Nas outras 3 hipóteses resultantes, (0 com 1), (1 com 0) ou (0 com 0) o resultado será 0 (reset).

EXEMPLO (em binário)

10101010 conteúdo do registro A
AND
11000000 o outro valor
Resultado
10000000 valor devolvido a A.

A instrução **AND A,** que não altera o conteúdo de A, é muitas vezes usada (já o temos feito) para colocar a **zero** o **Carry flag.**

Mnemónicas	Códigos
AND + N	230, + N
AND A	167
AND H	164
AND L	165
AND B	160
AND C	161
AND D	162
AND E	163
AND (HL)	166
AND (IX + d)	221, 166, + d
AND (IY + d)	253, 166, + d

O exemplo que segue é extraído da ROM do ZX81. A instrução AND + N vai ser usada para **mascarar de zero,** 2 bits do conteúdo de A.

A tabela das palavras chaves que se encontra nos endereços (273 a 507) tem memorizadas todas as palavras que se encontram no teclado. A última letra de cada uma está em **inverso de vídeo**, com a finalidade de servir de separador.

EXEMPLO: Palavra LIST

Endereços	Códigos	Letras
431	49	L
432	46	I
433	56	S
434	185	T

A rotina «Escreve uma palavra chave» começa por pesquisar a 1.ª letra da palavra, movendo o apontador (que neste caso é o registro BC) para o endereço 431; imprime todas as letras **regulares** (ou seja o L, I, S); decodifica o T em inverso de vídeo; imprime o T **normal** e efectua o retorno.

Resumindo:

Cada letra é AND com o binário 00111111 que não altera o código destas, mas mascara de zero a inversão de vídeo, transformando-a num código de letra normal.
No ZX81 o código de letra em inversão de vídeo, transformando-a num código de letra normal.

A operação AND, sobre as 4 letras:

Código 49 — 00110001 AND 00111111 = 00110001
» 46 — 00101110 AND 00111111 = 00101110
» 56 — 00111000 AND 00111111 = 00111000
» 185 — 10111001 AND 00111111 = 00111001

Após executada a instrução AND, cada letra é enviada ao impressor de caracteres (outra rotina da ROM) e o retorno é efectuado para dentro da nossa rotina «Escreve uma palavra/chave», aonde de novo o registro A vai ser carregado pelo código da letra apontada por BC, para ser testada se é ou não a última da palavra.
Este teste é executado dobrando o Código da letra, que sendo **normal** dá Carry (0) e, em inverso de video, Carry (1).
Se Carry = 1, a rotina efectua o retorno, porque a palavra já foi totalmente impressa.
Se Carry = 0, efectua-se um salto para o início da rotina, a fim de transportar a letra seguinte.

Endereços	Códigos	Mnemónicas	Comentários
2393	10	LD A, (BC)	cópia da letra
2394/5	230,63	AND + 63	mascara os últimos 2 bits
2396	Salto para a rotina impressora de caracteres		
2397	10	LD A, (BC)	volta a copiar a letra
2398	3	INC BC	aponta a letra seguinte
2399	135	ADD A, A	2 * o código
	Salto para 2393, se Carry (0)		
	Retorno se Carry (1)		

ENSAIO 1: (Spectrum) a utilização duma instrução AND para mascarar os bits que os atributos dão a cor do papel e da tinta.

Vamos usar um contador em Basic, que executará uma pequena rotina em C/M.

RAM TOP em 27999

Endereços	Mnemónicas	Códigos	Comentários
28000/2	LD A, (28020)	58, 116, 109	coloca em A o atributo

28003/4	AND + 241	230, 241	máscara dos bits 1, 2, 3 e 4
28005/6	ADD + 16	198, 16	mudança das cores
28007/9	LD (28020), A	50, 116, 109	envia atributo modificado
28010	RET	201	

BASIC:

```
10 BORDER 5: PAPER 5: INK 0: CLS
20 PRINT AT 10, 7; FLASH 1; INK 6; «ENSAIO AND»
40 FOR N = 22528 TO 23295 : REM AREA ATRIBUTOS
50 POKE 28020, PEEK N
60 RANDOMIZE USR 28000
70 POKE N, PEEK 28000
80 NEXT N
```

As cores do écran são mudadas de papel azul claro e tinta preta para amarelo e azul respectivamente.

Atributo inicial BIN (Papel CYAN/preta)	00101000
AND + 241	11110001
	00100000
ADD + 16	00010000
Atributo final (Papel amarelo e tinta preta)	00110000
Atributo inicial para as letras (Papel CYAN e tinta amarela, em flash)	10101110
AND + 241	11110001
	10100000
ADD + 16	00010000
Atributo final das letras (Papel amarelo e tinta preta em flash)	10110000

(Continua no próximo número)

HARDWARE

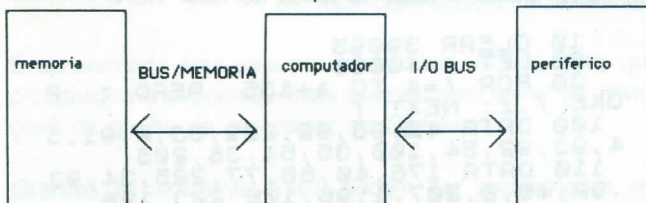
Autor: Alexandre Sousa

Iniciamos no número anterior o esboço de explicação dos diferentes métodos de implementar o interface ou seja o circuito de ligação entre os diferentes sectores de um computador.

No sistema em que é usado o método MEMORY-MAPPED I/O o interface comunica com o processador através do BUS da memória. Como resultado, o processador deixa menos espaço disponível para o utilizador, mas em contrapartida não necessita de instruções especiais para as entradas e saídas (I/O).

No segundo método de implementar um sistema de I/O, criamos um BUS completamente novo, a que chamamos I/O BUS, o qual se parece com o Memory BUS.

O bus de entrada/saída (I/O bus) tem um bus de endereço chamado peripheral-address bus (barramento de endereçamento dos periféricos) de modo a ser distinguido do barramento de endereçamento da memória.

Tem também um segundo conjunto de linhas de dados e um barramento de controlo de periféricos.

Os sinais no I/O bus podem ou não, ser parecidos com os do bus de memória. Este sistema tem a vantagem de possuir uma capacidade completa em termos do uso da memoria, mas tem a desvantagem de necessitar de um conjunto de instruções adicional chamado **I/O instructions**, um segundo barramento (I/O bus) também é considerado como desvantagem.

OBSERVAÇÃO: Um BUS ou BARRAMENTO não é mais do que um conjunto de linhas, através do qual se estabelecem as ligações entre dois circuitos electrónicos.

Devemos fazer uma breve pausa e passar uma vista de olhos pelo conjunto de instruções do processador.

O grupo principal que é denominado INSTRUÇÕES MÁQUINA, contém as instruções de manejo dos I/O e de referência à memória.

Esta classe de instruções controla (ao mais baixo nível) as operações do computador. Cada instrução pode iniciar apenas uma tarefa simples, tal como: obter um bit de informação a partir da memória ou endereçar uma linha de Entrada/Saída, ou ainda enviar um caracter para um dos dispositivos periféricos.

Aos programadores, está reservada a tarefa árdua, de resolver todos os problemas ligados ao projecto de programas a este nível de complexidade (programas em linguagem máquina).

Claro que o fabricante, dispõe sempre (para seu uso e dos programadores) de instrumentos — Sistemas Operativos — que propiciam um novo conjunto de operações com outro nível de potência.

Esse novo conjunto de instruções, denomina-se LINGUAGEM DE ALTO NÍVEL, porque essas instruções, agora referidas como ENUNCIADOS permitem programar ao mais alto nível (próximo à linguagem humana).

SINAIS DIGITAIS

Ao abordarmos esta discussão inicial, deixarmos claro (supomos) que no conjunto de linhas chamado BUS ou BARRAMENTO, circulam sinais, remetidos pelo processador e pelos outros componentes do sistema. Em qualquer linha do BUS podem ser medidos essas diferenças de tensão ou esses fluxos de corrente.

Um sinal que esteja presente num condutor, manifesta a sua existência pela presença ou ausência de tensão ou fluxo de corrente.

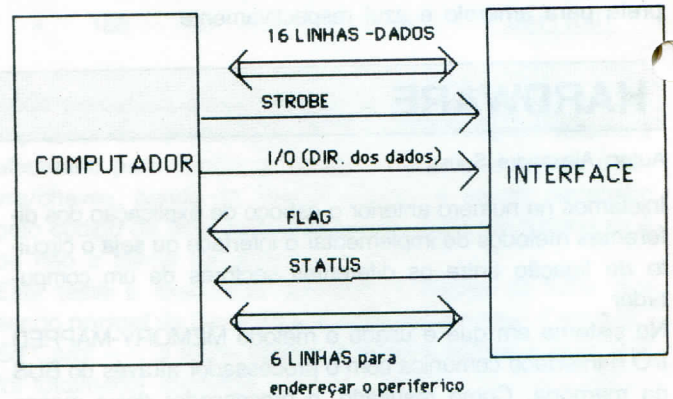
Será portanto o que designamos por SINAL DIGITAL ou BI-

NÁRIO porque ele assume sempre dois estados: PRESENTE ou AUSENTE.
Se tratamos de sinais relacionados por diferenças de tensão, existirá ou não essa diferença de tensão. A medida da voltagem será de K volts ou ZERO volts. As tensões são medidas em referência a um ponto zero, usualmente referido como GROUND ou MASSA ou TERRA, que não é mais do que um fio condutor comum a todos os circuitos e componentes existentes no sistema.
Os sinais binários ou digitais, são o meio primário de comunicação nos sistemas computadorizados, porque os circuitos de detecção ou os que geram os sinais binários, são simples de construir ou montar.
O que chamamos BUS não é mais do que um conjunto de fios condutores nos quais podem circular sinais digitais, dos quais o mais comum é conhecido como TTL (transístor-transístor-logic). Esta família de circuitos constitui um conjunto de blocos presentes em todos os computadores.
Estes circuitos não só definem a presença ou ausência de sinais binários como também definem os níveis ou regiões desses sinais.

Sinal 1 ou nível Alto .... 2 a 5 volts ..... High
Sinal 0 ou nível Baixo ..... 0.8 a 2 volts ..... Low
Sinal não definido .... 0 a 0.8 volts

Da mesma forma que possuimos circuitos para permitir a recepção ou envio de sinais a determinados níveis, também temos circuitos para transmitir sinais.
A tarefa principal dos circuitos do I/O é a de converter os níveis de sinal usados numa zona do sistema, noutro nível diferente, usado por outra zona. Por exemplo, os sinais usados nos sistemas de comunicações não são do mesmo nível que os sinais usados no processador.

REPRESENTAÇÃO DOS DADOS

Após o estabelecimento dos níveis dos sinais, precisamos de estabelecer o acordo sobre a representação dos vários sinais.
Qual a representação digital do caracter «A» ou do número «27»?
O alfabeto pode assumir 26 valores. Os numerais podem assumir um número infinito de valores. Como podemos representar todos estes valores apenas com 0 e 1 ou ON/OFF?
A resposta é a de que devemos usar mais do que uma linha de sinal, ou seja iremos criar um BUS. Se possuimos 8 linhas, cada uma das quais podem assumir dois níveis, poderemos representar 2 elevado à oitava potência ou seja 256 valores.

ROTINICES

Adapt.: J. MAGALHÃES/Clube Z80

ROTINA 1 — Ordenação alfabética/numérica

Agora, com esta rotina, pode ordenar o seu ficheiro por ordem alfabética e num período de tempo muito inferior àquele ocupado por qualquer outra em BASIC. Observe a figura 1 e compare os tempos.

DIM.	BASIC		CÓD. MÁQ.	
	Min.	Seg.	Min.	Seg.
(10,10)		3.5		0.04
(25,25)		18		0.38
(50,50)	1	28		3.04
(10,50)		3.5		0.1
(200,6)	27	30		7.18
(10,20)		3.5		0.06
(100,100)	7	5		24.18

FIGURA 1 — Tabela de tempos

A rotina ocupa 138 bytes de memória e pode ser colocada em qualquer posição. Geralmente, este género de rotinas funciona no «buffer» da Printer e, no caso de a desejar aí, lembre-se que tem de ficar no endereço além de 23310; assim, como sabe, não poderá usar instruções como LPRINT, COPY ou LLIST, pois elas fariam desaparecer a rotina.
Depois de introduzida a rotina (fig. 2), grave-a para evitar introduzi-la de novo em caso de erro, e então faça RUN. O código é posicionado (POKED) na memória, no endereço especificado na linha 10. Se pretender outra posição de memória, basta alterar a linha 10 antes de fazer RUN.

```
  10 CLEAR 39995
  20 LET a=40000
  30 FOR f=a TO a+135: READ z: P
OKE f,z: NEXT f
 100 DATA 42,93,92,229,33,4,91,3
4,93,92,54,100,35,54,36,205
 110 DATA 178,40,68,77,225,34,93
,92,48,2,207,1,96,105,203,126
 120 DATA 40,248,35,35,35,62,2,1
90,40,2,207,2,35,94,35,86,35,35
 130 DATA 175,190,32,244,43,126,
35,35,50,10,91,34,4,91,6,0,79,9
 140 DATA 34,6,91,27,66,75,237,6
7,8,91,197,42,4,91,237,91,6,91
 150 DATA 237,75,8,91,197,229,21
3,58,10,91,71,26,190,32,5,35,19
 160 DATA 16,248,175,209,225,58,
10,91,71,48,5,78,26,119,121,18
 170 DATA 35,19,16,245,193,11,12
0,177,32,217,193,11,120,177,32,1
99,201
```

FIGURA 2 — Rotina

Para usar esta rotina, os seus dados terão de entrar sob a forma de ARRAY D$. Este array deve ter 2 subscritos: D$ (a, b,), em que «a» corresponde ao número de subscritos (número das diferentes strings contidas no array) e «b» corresponde ao comprimento de cada subscrito. Este valor (de b) não deve exceder 255.

ERROS:

«Variable not found» — Este erro pode ocorrer quando D$ não existe na memória. «Subscript Wrong» — Se D$ está em memória, mas não dimensionado correctamente, como no exemplo D$ (4) ou D$ (4, 2, 2).

NOTA: Para ordenação numérica, deverá fazer um «loop» de modo a converter o array numérico numa string.

```
100 REM start
110 REM org 40000
120 REM
130 REM equ 23300   FIRST
140 REM equ 23302   SECOND
150 REM equ 23304   LOOP
160 REM equ 23306   SUB-LEN
170 REM equ 23681   STATUS
180 REM equ 23645   CH-ADD
190 REM equ &28B2   LOOK-VARS
200 REM
210 REM ld hl,(CH-ADD)
220 REM push hl
230 REM ld hl,FIRST
240 REM ld (CH-ADD),hl
250 REM ld (hl),100;'d'
260 REM inc hl
270 REM ld(hl),36;'$'
280 REM call LOOK-VARS
290 REM ld b,h
300 REM ld c,l
310 REM pop hl
320 REM ld (CH-ADD),hl
330 REM jr nc,FOUND
340 REM NOT-FOUND;
350 REM rst 8
360 REM datab &01
370 REM FOUND;
380 REM ld h,b
390 REM ld l,c
400 REM bit 7,(hl)
410 REM jr z,NOT-FOUND
420 REM inc hl
430 REM inc hl
440 REM inc hl
450 REM ld a,2
460 REM cp (hl)
```

```
470 REM jr z,IN-RANGE
480 REM ERROR;
490 REM rst 8
500 REM datab &02
510 REM IN-RANGE;
520 REM inc hl
530 REM ld e,(hl)
540 REM inc hl
550 REM ld d,(hl)
560 REM inc hl
570 REM inc hl
580 REM xor a
590 REM cp (hl)
600 REM jr nz,ERROR
610 REM dec hl
620 REM ld a,(hl)
630 REM inc hl
640 REM inc hl
650 REM ld (SUB-LEN),a
660 REM ld (FIRST),hl
670 REM ld b,0
680 REM ld c,a
690 REM add hl,bc
700 REM ld (SECOND),hl
710 REM dec de
720 REM ld b,d
730 REM ld c,e
740 REM ld (LOOP),bc
750 REM LOOP-A;
760 REM push bc
770 REM ld hl,(FIRST)
780 REM ld de,(SECOND)
790 REM ld bc,(LOOP)
800 REM LOOP-B;
810 REM push bc
820 REM push hl
830 REM push de
```

```
840 REM ld a,(SUB-LEN)
850 REM ld b,a
860 REM LOOP-C;
870 REM ld a,(de)
880 REM cp (hl)
890 REM jr nz,CONT
900 REM inc hl
910 REM inc de
920 REM djnz LOOP-C
930 REM xor a
940 REM CONT;
950 REM pop de
960 REM pop hl
970 REM ld a,(SUB-LEN)
980 REM ld b,a
990 REM LOOP-I;
1000 REM jr nc,NO-SWAP
1010 REM ld c,(hl)
1020 REM ld a,(de)
1030 REM ld (hl),a
1040 REM ld a,c
1050 REM ld (de),a
1060 REM NO-SWAP;
1070 REM inc hl
1080 REM inc de
1090 REM djnz LOOP-I
1100 REM pop bc
1110 REM dec bc
1120 REM ld a,b
1130 REM or c
1140 REM jr nz,LOOP-B
1150 REM pop bc
1160 REM dec bc
1170 REM ld a,b
1180 REM or c
1190 REM jr nz,LOOP-A
1200 REM ret
1210 REM end
```

FIGURA 3 — A mesma rotina «disassembled»

ROTINA 2 — Movimentos suaves

(in. YOUR COMPUTER, n.º 7, Setembro/84)

É apresentada aqui uma pequena rotina em cód. máq. que proporciona movimentos mais suaves sem sofrer as restrições da grelha de caracteres do Spectrum.

Gráficos e caracteres ASCII podem ser posicionados em qualquer coordenada de alta resolução com um simples comando BASIC. Você não fica limitado às 704 posições PRINT (0 - 21 linhas e 0 — 31 colunas), podendo posicionar os caracteres em qualquer ponto da grelha de alta resolução 256/176.

O programa utiliza apenas 120 bytes de memória, sendo completamente re-colocáveis, o que significa que você pode carregá-los em qualquer posição de memória.

Funciona sem quaisquer problemas no 16 K.

Outra vantagem em relação ao comando PRINT AT é que este permite-lhe usar mais 90 gráficos UDG *, a somar aos 21 (códigos 144-164).

Em suma, com esta rotina pode definir e posicionar caracteres de 165 a 255.

COMO CARREGAR A ROTINA?

(listagem 2)

A primeira parte, em BASIC, carregará o código máquina em qualquer posição de memória.

Introduza a listagem cuidadosamente (atenção às linhas DATA!). No Spectrum 48 K pode colocar o código, por exemplo, no endereço 64500. Faça CLEAR 64499, para **dizer** ao BASIC que não deve usar endereços superiores a 64499, e então RUN.

* UDG — User Defined Graphics (gráficos definidos pelo utilizador).

Introduza 64500 no endereço de carregamento. O programa «lê» as linhas DATA e coloca-as no endereço correspondente.

Se ocorreu qualquer erro (na introdução dos DATA), ser-lhe-á dada uma mensagem; corrija o erro e faça RUN de novo. Não teste a rotina antes de aparecer a mensagem: «Posição do caracter...». O código agora está pronto para ser usado. É boa ideia fazer uso dos dois comandos SAVE gravando toda a rotina e o código máquina separadamente, pois facilitará o trabalho em caso de erro.

No Spectrum 16 K (partindo do princípio de que não tem código máquina na memória), introduza o código máquina no endereço 31670. Faça CLEAR 31669 e então especifique o endereço para carregamento — 31670. É claro que o pode colocar noutra posição; no entanto não se esqueça de o proteger com o respectivo comando CLEAR.

A rotina possui um comando numa versão de Alta Resolução, que substitui.

PRINT INK 8; PAPER 8; OVER 1; AT Y; X; CHR$ c

por

RANDOMIZE X AND Y = C + USR a

onde «x» e «y» são as coordenadas horizontal e vertical do canto superior esquerdo do caracter; «c» é o código ASCII do caracter e «a» o endereço onde foi colocado o código máquina. Então

RANDOMIZE 0 AND 175 = 65 + USR 64500

posicionará a letra «A» (caracter 65) no canto superior esquerdo do écran — partindo do princípio de que o cód. máq. foi colocado no endereço 64500. O RANDOMIZE é uma «simulação» («dummy» no original) para guardar o resultado da chamada USR. Se no seu programa usa números «RANDOM», deve alterar RANDOMIZE por uma variável de transferência («dummy»), tal como

LET dummy = 0 AND 175 = 65 + USR 64500

Pode especificar as coordenadas ou os códigos do carácter com expressões tais como as variáveis ou números. Por exemplo:

RANDOMIZE xpos + xdir AND ypos — ydir = CODE «*» + USR move

Na rotina é usada uma técnica muito simples para conseguir as 3 expressões antes da chamada USR. Ocorrerá erro se houver mais ou menos do que 3 valores.

Se utilizar cálculos mais complicados do que a adição, subtracção, multiplicação ou divisão, pode precisar de colocar cada coordenada ou o código do caracter entre parêntesis para que a rotina os distinga.

Nesta rotina não lhe é permitido usar para a coordenada y valores inferiores a 7, desde que cada caracter contenha 8 linhas. Um caracter na coordenada 0,6 tem a linha mais baixa na coordenada y mais pequena. Aparecerá a mensagem de erro «Integer out of range» se usar coordenadas verticais menores que 7 ou maiores que 175.

PROGRAMA EXEMPLO (listagem 1)

Apresenta uma bola a rolar pelo écran em várias velocidades. Esta é posicionada usando «LET d = », e não RANDOMIZE já que a sequência dos números «RANDOM» não é constantemente recomeçada sempre que a bola se mova. (Repare na linha 330 e certifique-se que usou vírgulas).

O código máquina usa sempre o comando OVER 1 e assim qualquer caracter pode ser apagado sem ser destruido o fundo, simplesmente porque será logo recolocado o «desenho» no mesmo lugar.

O Spectrum usa normalmente códigos de caracteres entre 165 e 255 para representar palavras-chave tais como THEN, PRINT, ... que não têm necessidade de «saltar» no écran. Esta rotina permite-lhe definir, em seu lugar, mais 91 gráficos UDG podendo assim contar com 122 caracteres (juntando os 21 gráficos UDG standard).

Em princípio os novos caracteres são definidos exactamente do mesmo modo que os anteriores, ficando sequencialmente na memória. Isto significa que precisa expandir a área UDG antes de definir os gráficos que irão para além dos 21 standard.

No Spectrum 48 K, deverá utilizar os seguintes comandos para expandir essa área até 122 caracteres:

CLEAR 64559 : POKE 23675, 48 : POKE 23676, 252

Os POKES ajustam a variável de sistema CHARS para 728 (91 * 8) bytes adicionais. Como eles não reservam memória para código máquina é necessário o CLEAR 64500 e, por isso, terá de carregar a rotina no endereço 64500.

No 16 K carregue o código a 31670 e reserve o espaço com:

CLEAR 31669 : POKE 23675,48 : POKE 23676,124

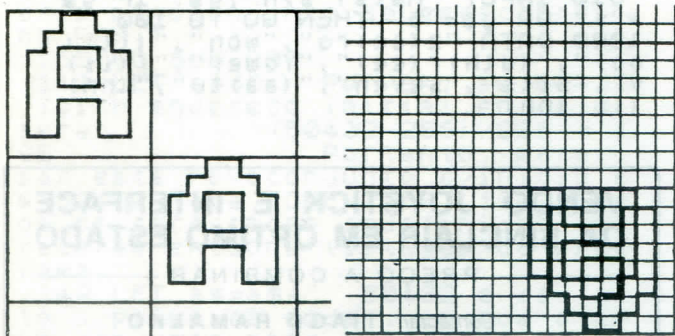
Estes gráficos assim definidos são colocados (poke) normalmente em memória; a única diferença é que não pode usar a função USR. Os caracteres extra seguem-se com intervalos dde 8 bytes. Os gráficos UDG correspondentes a «a» tem o código de caracter 144 e é possível encontrar a sua definição com o código «N», introduzindo:

PRINT USR «a» + 8 * (N-144)

Para melhor compreensão da utilidade desta rotina, passe a listagem 1 e grave-a, certificando-se de que não foram cometidos quaisquer erros. Ao fazer RUN, o programa ficará à espera da entrada da rotina em código máquina.

A rotina entra nomalmente como a anterior, apenas com uma pequena diferença na gravação.

Depois de completa e cuidadosamente introduzida, faça RUN. Prepare a cassette e o gravador para gravação e carregue numa tecla do Spectrum aquando da mensagem: «START TAPE AND PRESS ANY KEY». (Em caso de erro verifique de novo pela listagem).

Segue-se novamente a mensagem para gravação. Sem parar o gravador accione qualquer tecla do Spectrum. Terminada esta operação, tem duas gravações: a da rotina completa e a do código máquina. A primeira é apenas para permitir correcções sem o código máquina estar ainda instalado pois, nesse caso, não obteria a sua listagem. É exactamente esta 2.ª gravação que deve entrar quando à chamada (LOAD » » CODE) da 1.ª listagem.

MOVIMENTO EM PIXELES — Usando a rotina apresentada, obtém-se uma grelha de 176/256 pixeles em vez da normal colocação de um caracter. A 1.ª figura mostra o salto de um caracter em bloco. A 2.ª figura (à direita) apresenta o mesmo salto, mas de pixel em pixel, em 8 movimentos.

LISTAGEM 1 — Programa exemplo

```
 130 REM CARREGAR O CODIGO
 140 CLEAR 64499
 150 LOAD "Mover"CODE 64500
 160 REM determina area
 170 LET xmax=247
 180 LET ymax=168
 190 REM determina posicoes
 200 LET xpos=INT (RND*200)
 210 LET ypos=175
 220 LET xdir=INT (RND*6+1)
 230 LET ydir=-INT (RND*5+1)
 240 REM define a bola
 250 RESTORE
 260 FOR i=USR "a" TO USR "e"
 270 READ d
 280 POKE i,d
 290 NEXT i
 300 LET shape=144
 310 INK 6: PAPER 2: BORDER 5
 320 FOR i=0 TO 21
 330 PRINT AT i,0, INVERSE 1,
 350 NEXT i
 360 GO TO 480
 390 REM move a bola
 400 LET oldx=xpos
 410 LET oldy=ypos
 420 LET xpos=xpos+xdir
 425 IF xpos>1 THEN IF xpos<xmax
 THEN GO TO 440
 430 LET xpos=oldx: LET xdir=INT
 (RND*7+1)*-SGN xdir
 435 BEEP .03,15
 440 LET ypos=ypos+ydir
 445 IF ypos>7 THEN IF ypos<ymax
+7 THEN GO TO 460
 450 LET ypos=oldy: LET ydir=INT
 (RND*5+1)*-SGN ydir
 455 BEEP .03,30
 460 LET d=oldx AND oldy=shape+U
SR 64500
 470 LET shape=shape+1-4*(shape=
147)
 480 LET d=xpos AND ypos=shape+U
SR 64500
 490 GO TO 400
 590 REM definicao da bola
 600 DATA 60,66,135,143,143,159,
126,60
 610 DATA 60,66,193,225,249,253,
126,60
 620 DATA 60,126,249,241,141,225
,66,60
 630 DATA 60,126,191,159,135,131
,66,60
 640 DATA 0
```

LISTAGEM 2 — Rotina de movimento

```
 100 REM MOVIMENTO
 200 INPUT "Endereco";L
 210 LET C=0
 220 FOR i=L TO L+119
 230 READ d
 240 LET c=c+d
 250 POKE i,d
 260 NEXT i
 270 IF c<>13017 THEN PRINT "err
o nos DATA": STOP
 280 PRINT "Posicao do caracter
c em x,y com"
 290 PRINT "RANDOMIZE x AND y=c+
USR ";L
 300 PRINT '"gravar na sequencia
..."
 310 SAVE "rotina"
 320 SAVE "Mover"CODE L,120
 330 STOP
 400 DATA 42,101,92,229,235,42
 410 DATA 99,92,1,15,0,9
 420 DATA 237,82,40,2,207,25
 430 DATA 205,162,45,254,128,56
 440 DATA 11,71,214,144,56,19
 450 DATA 237,91,123,92,24,4
 460 DATA 237,91,54,92,38,0
 470 DATA 111,41,41,41,25,24
 480 DATA 6,205,56,11,33,146
 490 DATA 92,229,221,225,205,162
 500 DATA 45,103,229,205,162,45
 510 DATA 225,111,229,14,8,225
 520 DATA 37,229,36,197,68,77
 530 DATA 205,170,34,193,71,175
 540 DATA 176,221,126,0,40,17
 550 DATA 235,38,0,111,62,8
 560 DATA 144,71,41,16,253,235
 570 DATA 126,170,119,35,123,174
 580 DATA 119,221,35,13,32,213
 590 DATA 225,225,34,101,92,201
```

ROTINA 3 — Listar todo o programa !!!???

Esta rotina pode entrar com um MERGE no seu programa, de modo a obter listagens a partir de e até determinada linha, não sendo necessária a listagem completa.

```
9905 DEF FN p(x)=PEEK x+256*PEEK
 (x+1): DEF FN n(x)=256*PEEK x+P
EEK (x+1)
9907 LET prog=FN p(23635): LET v
ars=FN p(23627)
9910 CLS : PRINT "listar linhas"
: INPUT "1a linha="; LINE a$: IF
 NOT LEN a$ OR VAL a$<1 OR VAL a
$>9999 THEN GO TO 9910
9915 LET start=VAL a$: PRINT AT
2,0;"INICIO";start
9920 INPUT "ultima linha="; LINE
 a$: IF NOT LEN a$ OR VAL a$<1 O
R VAL a$>9999 THEN GO TO 9920
9930 LET stop=VAL a$: PRINT AT 2
,16;"Parar";stop
9935 IF prog=vars THEN STOP
9940 LET line=FN n(prog): PRINT
AT 4,0;"linha";line: LET length=
FN p(prog+2): IF line<start THEN
 LET prog=prog+length+4: PRINT A
T 4,16;"Saltando": GO TO 9935
9950 IF line>stop THEN STOP
9955 PRINT AT 4,16; INVERSE 1;"A
SSEMBLANDO": PRINT
9960 DIM l$(length+4): LET x=LEN
 STR$ line: LET l$(4-x+1 TO 4)=S
TR$ line
```

```
9965 LET z=5: FOR x=1 TO length-
1
9970 LET y=PEEK (prog+3+x): IF y
=14 THEN LET x=x+5: LET l$=l$( T
O LEN l$-6): GO TO 9980
9975 LET l$(z)=CHR$ y: LET z=z+1
9980 NEXT x: PRINT AT 4,16;"IMPR
ESSAO ": PRINT l$( TO z-1): LPRI
NT l$( TO z-1): LET prog=prog+4+
length: PRINT AT 6,0; OVER 1;l$(
 TO z-1): GO TO 9935
```

ROTINA 4 — Tens sintetizador de voz?

Para evitar uma segunda pessoa para ler, por exemplo, uma listagem de código máquina, introduza esta rotina e confira com o Spectrum.

```
  10 LET keys=0: LET delay=100
  20 DIM a$(10,12): FOR i=1 TO 1
0: READ a$(i): NEXT i
 100 INPUT "Endereco inicial ";s
tart
 110 INPUT "Final ";finish
 200 FOR i=start TO finish
 210 LET contents=PEEK i
 220 LET b$=STR$ contents
 230 PRINT i;" ";contents
 240 POKE 23692,255
 250 LET c$=""
 300 FOR j=1 TO LEN b$
 310 LET cd=CODE b$(1)
 320 LET c$=c$+a$(cd-47)
 330 IF c$(LEN c$)=" " THEN LET
c$=c$( TO LEN c$-1): GO TO 330
 340 LET b$=b$(2 TO )
 350 NEXT j
 360 LET s$=c$
 400 PAUSE delay
 410 NEXT i
 500 PRINT "Fim do boloco determ
inado"
 510 INPUT "Mais? s/n";y$: IF y$
="s" OR y$="S" THEN GO TO 100
1000 DATA "z(ee)ro","won","(tt)(
oo)","(dth)r(ee)","fower","f(ii)
v","siks","seven","(aa)tu","(nn)
(ii)(nn)"
```


MATEMÁTICA/RAÍZES

SPECTRUM

Autor: CARLOS M. B. MORENO
Porto

Visto que o Spectrum não tira raízes de índice diferente de dois envio este programa que permite tirar raízes de qualquer índice e de qualquer número (mesmo com virgulas). No entanto devido ao processo utilizado este programa torna-se muito lento ao trabalhar com uma grande parte decimal.

```
  10 DIM r$(100)
  20 PRINT AT 0,0;"r": PLOT 0,16
6: DRAW 10,0: DRAW 5,-20: DRAW 5
,25: DRAW 50,0
  25 PRINT AT 2,5;"n"
  60 INPUT "r ";r
  70 INPUT "n "; LINE n$
  71 FOR e=1 TO LEN n$: IF n$(e)
="." THEN GO TO 75
  72 NEXT e
  73 LET n$=n$+"."
  75 LET n$=n$+"0000000000000000
00000000000000000000000000000000
0"
  76 CLS
  80 LET b$=STR$ LEN STR$ INT VA
L n$
  90 LET b=VAL b$
 100 FOR n=0 TO 4e4
 110 IF VAL STR$ (n↑r)>VAL n$(1
TO b) THEN GO TO 130
 120 NEXT n
 130 LET n=n-1
 135 LET b=b+2
 140 LET r$(1 TO LEN STR$ n)=STR
$ n
 150 LET rn=LEN STR$ n
 160 LET r$(rn+1)=".": LET rn=rn
+1
 170 FOR n=0 TO 9
 180 IF VAL (r$(1 TO rn)+STR$ n)
↑r>VAL n$(1 TO b) THEN GO TO 20
 190 NEXT n
 200 LET n=n-1
 210 LET b=b+1
 220 LET rn=rn+1: LET r$(rn)=STR
$ n
 225 PRINT AT 0,0;r$(1 TO rn)
 227 IF rn=11 THEN STOP
 230 GO TO 170
```

DEFINIDOR DE CARACTERES

SPECTRUM

Trad. e Adapt.: MÁRIO OLIVEIRA
Porto

```
in:Personal computing Today
Novembro 1984
Trad. e adapt.: Mario Oliveira

 100 POKE 23658,0: POKE 23607,60
: CLEAR 5e4: BORDER 0: PAPER 0:
INK 7: OVER 0: CLS
 102 LET j$=" !"#$%&'()*+,-./01
23456789:;<=>?@ABCDEFGHIJKLMNOPQ
RSTUVWXYZ[\]↑_£abcdefghijklmnopq
rstuvwxyz{|}~©"
 105 INPUT "Quer explicacoes ? "
; LINE y$: IF y$="" THEN GO TO 1
05
 107 IF y$(1)<>"s" AND y$(1)<>"S
" THEN GO TO 210
 110 LET a$="   Este programa pe
rmite dese   nhar 1152 graficos
em blocos de 96 de cada vez,para
 uso nos seusprogramas.
                  Primeiro, e feit
o o POKE dos dados do conjunto d
e caracteres da ROM (enderecos 1
5616 a 16383)para a RAM. Pode en
```

```
tao definir- se qualquer quantid
ade de simbo los, letras ou nume
ros, na RAM."
 120 GO SUB 200
 130 LET a$="  Este novo conjun
to de caracteres pode ser adopta
do fazendo           POKE 236
07,N         sendo N o endereco
inicial do novo conjunto menos 2
56 e divididopor 256"
 132 LET a$=a$+"
              23607 e a variavel CHAR
S onde e feito o POKE para apont
ar o novoconjunto de caracteres
na RAM."
 135 GO SUB 200
 140 LET a$="   Por ex.,se 50432
 foi o endereco inicial,entao N
sera:          (50432-256)/256 = 1
96                Portanto, para u
sar este novoconjunto c/inicio e
m 50432,faca POKE 23607,196. Tod
os os caracteres redefinidos apa
recerao entaona listagem do prog
rama."
 142 LET a$=a$+"  50432 e de fac
to o primeiro ender. em que o no
vo conjunto esta guarda do. Isto
 e,dispoe ainda de 27K+ para o r
esto do seu programa."
 145 GO SUB 200
 150 LET a$="   Se quiser voltar
 ao conjunto normal de caract.(l
ocalizado em 15616), entao N ter
a o valor de (15616-256)/256 = 6
0                 Logo,para regres
sar ao normaltera que fazer POKE
 23607,60"
 151 LET a$=a$+"
                    Depois de gr
avado, cada conjuntoPODE SER CAR
REGADO EM QUALQUER  ALTURA (mesm
o apos ter feito NEWdeste progra
ma. Basta para isso fazer LOAD "
"""CODE e depois POKE 23607,N (n
umero que tera de ano tar para f
uturas referencias)."
 155 GO SUB 200
 160 LET a$="  !NOTAS SOBRE O M
ODO DEFINICAO
                  Se escolher o mo
do definidor de caracteres, surg
irao 4 indicacoes em INVERSO VID
EO:          CHARS.SET:e o conj.
caracteres               que esta
a definir.   ENDERECO :e o end.
do primeiro               byte do c
onjunto."
 165 LET a$=a$+"         N :e o nume
ro cujo POKE deve ser   feito em
 23607, quando quiser   usar o n
ovo conjunto de car. REDEFINIR:e
 o conjunto de caract          q
ue esta a redefinir."
 170 GO SUB 200
 180 LET a$="  !NOTAS SOBRE O M
ODO DEFINICAO
                  Em resumo:
           ::
 184 LET a$=a$+"   Surgirao 2 gr
upos de caracte res ao fundo do
ecran, sendo um em INVERSO VIDEO
 e que servira de referencia. N
o outro conjuntoaparecerao os ca
racteres redem definidos."
 190 GO SUB 200
 199 GO TO 210
 200 CLS : PRINT AT 0,4; PAPER 1
; INK 6;"REDEFINICAO DE CARACTER
ES"'''''': FOR f=1 TO LEN a$: PR
INT a$(f);: IF a$(f)<>" " THEN B
EEP .01,40
```

```
 202 NEXT f: PRINT #0;"
Tecle uma letra          ": PAUSE
0: BEEP .5,20: RETURN
 210 CLS : INPUT "Nr. de conjunt
os a redefinir? "; LINE b$: IF b
$="" OR CODE b$<48 OR CODE b$>57
 THEN GO TO 210
 211 LET b=VAL b$: IF b<1 OR b>1
2 THEN GO TO 210
 215 LET po=60: BEEP .5,22
 220 PRINT AT 0,0; FLASH 1; BRIG
HT 1;"Espere, por favor";''''''
;"Estou ocupado a fazer POKES...
"
 225 DIM c(12): DIM h(8)
 226 LET h(1)=128: FOR f=2 TO 8:
 LET h(f)=h(f-1)/2: NEXT f
 230 FOR f=196 TO 196+(b-1)*4 ST
EP 4: LET c(f/4-48)=1: FOR g=156
16 TO 16383: POKE f*256+256+g-15
616,PEEK g: NEXT g: NEXT f
 240 BORDER 0: PAPER 0: INK 7: C
LS : PRINT TAB 14;"M E N U": PRI
NT AT 8,4;"1 > Definir novo conj
unto";AT 10,4;"2 > Carregar um c
onjunto";AT 12,4;"3 > Gravar um
conjunto";AT 14,4;"4 > Chamar um
 conjunto"
 250 LET i$=INKEY$: IF i$<"1" OR
 i$>"4" THEN GO TO 250
 260 GO TO 1000*VAL i$
1000 BEEP .5,20: BEEP .3,24: PAP
ER 6: OVER 0: BORDER 7: INK 1: C
LS : PRINT AT 0,0;"1 > DEFINIDOR
"
1002 FOR f=1 TO 6: POKE USR "a"+
f,129: NEXT f: POKE USR "a",255:
 POKE USR "a"+7,255
1005 POKE 23607,60: PRINT AT 15,
0; INVERSE 1;j$(1 TO 32)''j$(33
TO 64)''j$(65 TO 96)
1010 INPUT "Conjunto a redefinir
 ? "; LINE z$: IF z$="" OR CODE
z$<48 OR CODE z$>57 THEN GO TO 1
010
1011 LET ch=VAL z$: IF ch>12 OR
ch<1 THEN BEEP .5,-30: GO TO 101
0
1012 IF NOT c(ch) THEN BEEP .5,-
30: GO TO 1010
1015 PRINT AT 10,0; INVERSE 1;"C
ONJ.CARACT:"; INVERSE 0;ch
1016 LET ch=196+(ch-1)*4
1018 PRINT AT 11,0; INVERSE 1;"E
NDERECO   :"; INVERSE 0;ch*256+2
56
1022 POKE 23607,ch: PRINT AT 16,
0;j$(1 TO 32)''j$(33 TO 64)''j$(
65 TO 96)
1033 POKE 23607,po: INPUT AT 0,0
;"Caracter a redefinir ? "; LINE
 c$: IF CODE c$>127 OR CODE c$<3
2 OR LEN c$>1 THEN BEEP .5,-20:
GO TO 1033
1034 PRINT AT 13,0; INVERSE 1;"A
 REDEFINIR:"; INVERSE 0;c$;AT 12
,0; INVERSE 1;"          N :"; IN
VERSE 0;ch
1035 FOR f=1 TO 8: PRINT AT f,0;
"□□□□□□□□|□□□□□□□□": NEXT f
1036 PRINT AT 0,0; INK 3; PAPER
7;"COMPARAR|87654321"
1037 LET x=1: LET y=9: DIM a(8)
1038 DIM b$(8,8): LET k=(CODE c$
-32)*8
1040 PRINT AT 0,18;"->Flechas pa
ra";AT 1,20;"mover.";AT 2,18;"p=
plot.";AT 3,18;"e=apagar";AT 4,1
8;"c=fixar graf."
1041 PRINT AT 5,18;"n=fixar graf
.";AT 6,20;"e mudar";AT 7,20;"co
nj.car";AT 8,18;"o=comparar";AT
```

```
9,20;"grafico.";AT 10,18;"q=sobr
epor";AT 11,20;"grafico";AT 12,1
8;"m=aceder menu"
1042 PLOT 135,104: DRAW 0,-38: D
RAW 120,0
1045 OVER 1: INK 1: PRINT AT x,y
;"█": BEEP .003,x+y: PRINT AT x,
y;"█"
1046 LET i$=INKEY$
1050 LET y=y+(i$="8" AND y<16)-(
i$="5" AND y>9): LET x=x-(i$="7"
 AND x>1)+(i$="6" AND x<8)
1060 IF i$="p" AND b$(x,y-8)=" "
 THEN PRINT AT x,y; OVER 0;"█":
LET b$(x,y-8)="█": LET a(x)=a(x)
+2↑(16-y)
1065 IF i$="e" AND b$(x,y-8)="█"
 THEN PRINT AT x,y; OVER 0;"□":
LET b$(x,y-8)=" ": LET a(x)=a(x)
-2↑(16-y)
1070 IF i$="o" THEN GO SUB 1500
1075 IF i$="c" THEN GO SUB 1200:
 GO SUB 1700: GO TO 1030
1080 IF i$="n" THEN GO SUB 1200:
 GO SUB 1700: GO TO 1010
1085 IF i$="q" THEN GO SUB 1600
1090 IF i$="m" THEN CLS : BORDER
 0: PAPER 0: OVER 0: INK 7: GO T
O 240
1100 GO TO 1045
1200 OVER 0: FOR f=1 TO 8: POKE
ch*256+256+k+f-1,a(f): NEXT f: R
ETURN
1500 INPUT "Car. p/comparar ? ";
 LINE g$: IF CODE g$<32 OR CODE
g$>127 OR LEN g$>1 THEN BEEP .5,
-20: GO TO 1500
1505 FOR f=1 TO 8: PRINT AT f,0;
 OVER 0;"□□□□□□□□": NEXT f
1510 FOR f=0 TO 7: LET gr=PEEK (
ch*256+256+((CODE g$-32)*8)+f)
1515 FOR g=1 TO 8
1520 IF gr>=h(g) THEN PRINT AT 1
+f,g-1; OVER 0;"█": LET gr=gr-h(
g)
1530 NEXT g: NEXT f
1540 RETURN
1600 INPUT "Car. p/sobrepor ? ";
 LINE g$: IF CODE g$<32 OR CODE
g$>127 OR LEN g$>1 THEN BEEP .5,
-20: GO TO 1600
1605 FOR f=1 TO 8: PRINT AT f,9;
 OVER 0;"□□□□□□□□": NEXT f
1607 DIM a(8)
1610 FOR f=0 TO 7: LET gr=PEEK (
ch*256+256+((CODE g$-32)*8)+f)
1615 FOR g=1 TO 8
1620 IF gr>=h(g) THEN PRINT AT 1
+f,g+8; OVER 0;"█": LET gr=gr-h(
g): LET a(f+1)=a(f+1)+h(g): LET
b$(f+1,g)="█"
1630 NEXT g: NEXT f: RETURN
1700 LET k=CODE c$-32: IF k<32 T
HEN LET xx=16: GO TO 1730
1710 IF k<64 THEN LET xx=18: LET
 k=k-32: GO TO 1730
1720 IF k<96 THEN LET xx=20: LET
 k=k-64
1730 POKE 23607,ch: PRINT AT xx,
k;c$: POKE 23607,po: RETURN
2000 BEEP 1,10: BORDER 6: PAPER
6: INK 0: OVER 0: CLS : PRINT AT
 0,0;"2 > CARREGAR"
2010 PRINT AT 4,1;"Nome do conju
nto ?              (Se desconhec
er, tecle ENTER)"
2020 INPUT d$
2030 PRINT AT 7,0;: LOAD d$CODE
2040 FOR f=196 TO 244 STEP 4: IF
 NOT PEEK (f*256+255) THEN NEXT
f
2045 POKE f*256+255,0
2050 PRINT AT 8,0;"Carregou o co
nj. ";f/4-48;AT 10,0;"Para o cha
mar no seu programa   faca POKE
23607,";f;AT 13,0;"Este conjunto
 localiza-se em ";f*256+256
2055 LET c(f/4-48)=1
2060 PRINT AT 16,0;"CONJ.CARACT:
";f/4-48: POKE 23607,f: PRINT j$
: POKE 23607,po
2070 PRINT #0;" TECLE QUALQUER
LETRA P/ MENU ": PAUSE 0: BEEP
.2,13: BEEP .3,11: GO TO 240
3000 BEEP 1,12: BORDER 4: PAPER
6: INK 0: OVER 0: CLS : PRINT AT
 0,0;"3 > GRAVAR UM CONJUNTO"
3010 INPUT "Conjunto a gravar ?
"; LINE c$: IF CODE c$<48 OR COD
E c$>57 THEN BEEP .5,-30: GO TO
3010
3012 LET c=VAL c$: IF c<1 OR c>1
2 THEN BEEP .5,-40: GO TO 3010
3013 IF NOT c(c) THEN BEEP .6,-2
2: GO TO 3010
3020 PRINT AT 2,0;"A gravar o co
njunto ";c$
3025 LET s$="Chars. "+STR$ c
3027 LET c=(c+48)*4
3030 POKE c*256+255,1: SAVE s$CO
DE c*256+255,769
3040 BEEP .6,2: PRINT AT 4,0;"A
verificar": PRINT AT 6,0;: VERIF
Y s$CODE : BEEP .8,5: PRINT AT 7
,0;"Codigo verificado"
3045 PRINT AT 10,0;"Quer gravar
de novo ? ": INPUT y$: IF y$(1)=
"s" THEN GO TO 3027
3048 PRINT AT 10,0;"Quer gravar
outro conjunto ? ": INPUT y$: IF
 y$(1)="s" THEN GO TO 3e3
3050 GO TO 2070
4000 BEEP 1,22: PAPER 5: BORDER
6: INK 0: OVER 0: CLS : PRINT AT
 0,0;"4 > CHAMADA DO CONJUNTO"
4010 INPUT "Que conj. quer traba
lhar ? "; LINE c$: IF CODE c$<48
 OR CODE c$>57 THEN BEEP .5,-35:
 GO TO 4010
4012 LET b=VAL c$: IF b=0 THEN L
ET po=60: POKE 23607,po: GO TO 2
070
4015 IF b<1 OR b>12 THEN BEEP .5
,-22: GO TO 4010
4016 IF NOT c(b) THEN GO TO 4010
4020 PRINT AT 2,0;"Se em qualque
r momento quiser   regressar a c
onfiguracao normal de caracteres
, seleccione esta  opcao e indiq
ue que quer traba  lhar o conjun
to '0'.            Nao esqueca q
ue os caracteres jaredefinidos a
pareceerao na lista gem do progra
ma!"
4030 LET po=(b+48)*4: POKE 23607
,po: GO TO 2070
```

NOTA: Caracteres linhas 1505 e 1605 - USR «A»

JETSET WILLY

SPECTRUM

Desde o aparecimento de JET SET WILLY que várias pessoas se esforçaram para descobrir o POKES que facilitam o seu uso.
Este programa contém alguns desses POKES. Depois de o introduzir, grave-o antes do último bloco do JET SET WILLY.

```
  5 CLEAR 32767: LOAD ""CODE
 10 CLS : PRINT "1-vidas ilimit
adas"
 20 PRINT "2-escolher numero de
 vidas"
 30 PRINT "3-escolher o numero
de objectos"
 40 PRINT "4-tirar o problema d
o ATTIC"
 50 PRINT "5-tirar os bonecos"
 52 PRINT "6-ficar invulneravel
"
 54 PRINT "7-nao morrer ao cair
 muito"
 56 PRINT "8-escrever o WRITETY
PER"
 58 PRINT "9-simplificar o BANY
AN TREE"
 60 PRINT "0-dar saltos gigante
scos"
 62 PRINT "S-passar a imagem do
 visor para o gravador durante o
 jogo,basta premir a tecla s"
 64 PRINT "I-permitir um funcio
namento      correcto com o Inter
face 1       ligado"
 66 PRINT "J-terminar"
 68 IF INKEY$="1" THEN POKE 358
99,0
 70 IF INKEY$="2" THEN INPUT "n
umero de vidas <32 ";v: POKE 347
85,v
 72 IF INKEY$="3" THEN INPUT "n
umero de objectos ";n: POKE 4198
3,256-n
 74 IF INKEY$="4" THEN POKE 599
00,255
 76 IF INKEY$="5" THEN POKE 351
23,0
 78 IF INKEY$="6" THEN POKE 348
08,0: POKE 34809,0: POKE 34811,0
: POKE 34812,0: POKE 34814,0: PO
KE 34815,0: POKE 37425,7: POKE 4
0064,0: POKE 40191,0
 80 IF INKEY$="7" THEN POKE 364
77,1
 82 IF INKEY$="8" THEN POKE 342
75,10
 84 IF INKEY$="9" THEN POKE 365
45,0
 86 IF INKEY$="0" THEN POKE 363
58,0
 88 IF INKEY$="s" OR INKEY$="S"
 THEN RESTORE 88: POKE 35538,191
: POKE 35600,14: POKE 35601,254:
 POKE 34997,0: POKE 34998,0: POK
E 34999,0: FOR i=35547 TO 35590:
 READ a: POKE i,a: NEXT i: DATA
253,229,221,229,221,33,237,255,1
7,17,0,175,205,194,4,6,50,118,16
,253,17,0,27,62,255,221,33,0,64,
205,194,4,243,221,225,253,225,14
,254,0,0,0,0,0
 90 IF INKEY$="i" OR INKEY$="I"
 THEN RESTORE 90: POKE 35591,195
: POKE 35592,240: POKE 35593,255
: FOR i=65520 TO 65535: READ a:
POKE i,a: NEXT i: DATA 197,33,0,
154,17,0,90,1,0,1,237,176,193,19
5,16,139
 95 IF INKEY$="j" OR INKEY$="J"
 THEN RANDOMIZE USR 33792
100 GO TO 68
```

NÚMEROS PRIMOS

NEWBRAIN

Autor: PAULO CASTELO
Porto

```
10 REM:   ***************************************************************
20 REM:   **                                                           **
30 REM:   **  NUMEROS PRIMOS   V:5.0     "*10:P5"        NB4807181I   **
40 REM:   **                                                           **
50 REM:   **  QU.18/7/84     Paulo Castelo      BASIC   NewBrain AD   **
60 REM:   **                                                           **
70 REM:   ***************************************************************
80
90
100 REM ------------------------------------------[ Inicializacao dos Canais ]
110 REM:
120 OPEN#0,0,"1"  : CLOSE#8 : OPEN#8,9,"2400"
130 PUT#8,27,"L",4 : PRINT#8, 2[7];3[6];5[6];"      7";
140
150 REM ---------------------------------------------------------[ Comentarios ]
160 REM:
170 REM:.....................  PRINT A [6] ; ....... PRINT USING "-###### "; A;
180 REM:...  PRINT MID$( STR$( P[8] ),2,8); ....... PRINT USING "########"; P;
190
200 REM ---------------------------------------[ Inicializacao das Variaveis ]
210 REM:
220 CLEAR : OPTION BASE 1 : N=4500
230 DIM P(N) :  P(1)=2 : P(2)=3 : P(3)=5 : P(4)=7 :  RQ=4 : I=4 :  L=3 : F=0
240 PUT 31, 23,2, 151
290
300 REM -------------------------------------------[ Geracao dos N.s Primos ]
```

```
310 REM:
320 P=7  : GOTO 340
330    P=P+6 : GOSUB 450
340    P=P+4 : GOSUB 450
350    P=P+2 : GOSUB 450
360    P=P+4 : GOSUB 450
370    P=P+2 : GOSUB 450
380    P=P+4 : GOSUB 450
390    P=P+6 : GOSUB 450
400    P=P+2 : GOSUB 450      REM: Esta sequencia faz com que os N.s nao
410 GOTO 330                  REM: sejam multiplos de 2,3 e 5.
440
450 REM -------------------------------------------------[ Teste se P e' Primo ]
460 REM:
470 EP= TRUE
480 FOR J=4 TO RQ
490     T=P/P(J)
500     IF T=P(J) AND J=RQ THEN RQ=RQ+1
510     IF T=INT(T) THEN J=RQ : EP= FALSE
520 NEXT J
530 IF EP GOSUB 600 : REM:  E' Primo.
540 RETURN
590
600 REM ------------------------------------------[ Procedimento para P  Primo ]
610 REM:
620 L=L+1  : IF L=10 PRINT#8  : L=0  : F=F+1
630          IF F=50 PUT#8, 13,13,13,13  : F=0
640 I=I+1  . IF I>N THEN 660
650 P(I)=P
660 PRINT#8, MID$( STR$( P[8] ),2,8 );
670 PRINT CHR$(12); I[10]; ":"; P[10];
680 RETURN
```

BASIC EXPANSION/SPECTRUM COM INTERFACE 1

Adapt.: MANUEL QUINAZ
Porto

```
  1 REM Basic expansion
  2 REM Novos Comandos-

 1. DRAW Absoluto
    *draw 120,130
 2. PRINT em alta resolucao
    *print x,y,c
  (x,y)=coordenadas ,c=UDG a ser
imprimido,A=1,B=2,etc.
 3. SCROLL
    *scroll n
    n=0>  n=1↑  n=2<  n=3v
 4. SONS
    *zap n
    n varia entre 0 e 2
  10 CLEAR 64797
  20 LET start=64798
  30 LET a$=""
  40 FOR a=start TO start+558
  50 IF a$="" THEN READ a$
  60 POKE a,FN h(a$)*16+FN h(a$(
2))
  70 LET a$=a$(3 TO )
  80 NEXT a
 100 LET a=20: LET p=45: GO SUB
500
 110 LET a=53: LET p=99: GO SUB
500
 120 LET a=77: LET p=165: GO SUB
500
 130 LET a=83: LET p=267: GO SUB
500
 140 LET a=97: LET p=465: GO SUB
500
 150 LET a=100: LET p=31: GO SUB
500
 160 LET a=166: LET p=31: GO SUB
500
 170 LET a=268: LET p=31: GO SUB
500
 180 LET a=463: LET p=394: GO SU
B 500
 190 LET a=466: LET p=31: GO SUB
500
 200 PRINT "To use the extra com
mands, enter"
 210 PRINT "POKE 23735,";start-
256*INT (start/256);": POKE 2373
6,";INT (start/256)
 220 STOP
 500 LET address=start+a+1
 510 LET pointsto=start+p
 520 POKE address,INT (pointsto/
256)
 530 POKE address-1,pointsto-256
*PEEK address
 540 RETURN
 900 DEF FN h(a$)=CODE a$-48-7*(
a$>="A")
1000 DATA "D71800FE2AC2F001D720"
1010 DATA "00E69FFE1B3801AF8721"
1020 DATA "2DD806004F095E2356EB"
1030 DATA "E9D77400FE20C8FE3AC8"
1040 DATA "FE0D20F3C9F001F001F0"
1050 DATA "01F00163D8F001F001F0"
1060 DATA "01F001F001F001F001F0"
1070 DATA "01F001F001F001A5D8F0"
1080 DATA "01F0010BD9F001F001F0"
1090 DATA "01F001F001F001D1D9CD"
1100 DATA "1FD8D7821CFE2CC2F001"
1110 DATA "D72000D7821CCDB705D7"
1120 DATA "941E217E5C9638141601"
1130 DATA "47C5D5D7941ED1C1217D"
1140 DATA "5C96380A1E01180A16FF"
1150 DATA "ED4418E81EFFED444FD7"
1160 DATA "BA24C3C105CD1FD8D782"
1170 DATA "1CFE2CC2F001D72000D7"
1180 DATA "821CFE2CC2F001D72000"
1190 DATA "D7821CCDB705D7941ECB"
1200 DATA "27CB27CB2716005F2A7B"
1210 DATA "5C19E5D7941EF5D7941E"
```

```
1220 DATA "C14FD13E08F5C5D7AA22"
1230 DATA "F5E5D5D74D0DD7DB0BE1"
1240 DATA "46D1EBF10E003CCB38CB"
1250 DATA "193D20F9702371C10513"
1260 DATA "F13D20D9C3C105CD1FD8"
1270 DATA "D7821CCDB705D7941EE6"
1280 DATA "03280AFE01282CFE0228"
1290 DATA "15186C3EC0210040A706"
1300 DATA "20CB1E2310FB3D20F5C3"
1310 DATA "C1053EC021FF57A70620"
1320 DATA "CB162B10FB3D20F5C3C1"
1330 DATA "05A71100400603C53E08"
1340 DATA "083E07626B24E5012000"
1350 DATA "EDB0D13D20F3010007ED"
1360 DATA "42E5012000EDB0D1083D"
1370 DATA "20E001E00609545D0120"
1380 DATA "00ED42EBEDB0C110CC21"
1390 DATA "E05706207723 10FCC3C1"
1400 DATA "0511FF570603C53E0808"
1410 DATA "3E07626B25E5012000ED"
1420 DATA "B8D13D20F301000709E5"
1430 DATA "012000EDB8D1083D20E1"
1440 DATA "01E006ED42545D012000"
1450 DATA "09EBEDB8C110CD210040"
1460 DATA "0620C38AD9CD1FD8D782"
1470 DATA "1CCDB705D7941EA72832"
1480 DATA "FE012809FE02280DFD36"
1490 DATA "000AEF01800121100018"
1500 DATA "0601FFFE212018110400"
1510 DATA "3E10C5D5E5F5D7B503F1"
1520 DATA "E1D1C1093D20F1C3C105"
1530 DATA "F33A485C0F0F0F260446"
1540 DATA "2B10FED3FEEE10087CB5"
1550 DATA "28030818F0FBC3C105"
```

COLDITZ

SPECTRUM

Autor: ARMANDO BESSA
Bragança

Pequeno jogo que põe à prova os teus reflexos.
Deslocando o cursor da sua posição inicial (teclas 5, 6, 7, 8) deves controlá-lo de forma a percorreres toda a grelha até ao ponto de chegada assinalado em FLASH.
Repara que tens apenas uma saída e o tempo é factor fundamental da tua pontuação.

```
  1 LET u=9000: DIM a$(20*32)
  9 REM ...labirinto
 10 LET s$="--FUGA DE COLDITZ--
-": RANDOMIZE : FOR f=0 TO 19: L
ET a$(f*32+1 TO f*32+32)="■ ■ ■
■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■"+s$(f+
1): NEXT f: FOR f=3 TO 31 STEP 2
: LET x=INT (RND*15)+3: LET a$(f
+x*32)=" ": NEXT f
 20 PRINT AT 0,0; INVERSE 1;"##
#
 INVERSE 0; PAPER 4;a$;"■■■■■■■■
■■■■■■■■■■■■■■■■■■■■■■■■";AT x+1
,30; PAPER 2; INK 6; FLASH 1;"*"
 30 BORDER 1: INPUT ""
 40 LET a=1: LET b=1
 50 PRINT AT 1,0; INVERSE 1;"#+
#";AT 2,0;"###"
 60 PAUSE 0
 65 REM ...tempo a zero
 70 LET s=23672: POKE s,0: POKE
s+1,0: POKE s+2,0
 90 REM ...ciclo do jogo
100 PRINT AT a,b;"+"
105 LET a1=a: LET b1=b
110 LET x=PEEK s+PEEK (s+1)*256
+PEEK (s+2)*65536
120 PRINT #0;AT 0,0; PAPER 2; B
RIGHT 1;"TEMPO: ";x
130 IF b=30 THEN GO TO 200
135 REM ...define as teclas
140 LET a=a+(INKEY$="6" AND a<2
0)-(INKEY$="7" AND a>1)
150 LET b=b-(INKEY$="5")+(INKEY
$="8")
155 REM ...ve se ha um obstaclo
160 IF A$((a-1)*32+b+1)="■" THE
N LET a=a1: LET b=b1
170 IF a=a1 AND b=b1 THEN GO TO
100
180 PRINT AT a1,b1; PAPER 4;" "
190 GO TO 100
195 REM ...fim
200 BEEP 2,-60: BEEP .02,20: PR
INT #0;"MELHOR ATE AGORA-";u
210 IF x<u THEN PRINT #0;AT 0,1
2; FLASH 1; INK 0; PAPER 7;"->NO
VO MAXIMO": LET u=x
220 PAUSE 0
240 CLS : GO TO 10
```

ACHOU/GANHOU

SPECTRUM

Adapt.: HUGO ASSUNÇÃO
Lisboa

Educacional. Teste para os mais pequenos, resolução de operações.
Quanto mais rápido for a dar a resposta, maior será a percentagem final.

```
  0>REM ZX COMPUTING Jn/Jl 84
        Adaptado por H.A.
  5 FOR G=15 TO 21: PRINT AT G,
0;,,: NEXT G: RETURN
 10 REM ***ACHOU/GANHOU***
 20 FOR n=1 TO 4
 30 READ d
 40 FOR m=1 TO d
 50 READ a: BEEP .02,a*RND: PRI
NT AT n,a; FLASH 1;"*"
 60 NEXT m
 70 NEXT n
 80 RETURN
 85 REM ***COMECO***
 90 INK 0: PAPER 6: BORDER 6: C
LS
100 INK 2: PAPER 4
110 GO SUB 20
120 INK 6: PAPER 4
130 GO SUB 20: PRINT AT 0,30; F
LASH 1;" "
140 INK 2: PAPER 6
150 GO SUB 20: PRINT AT 6,29; F
LASH 1;" ": FLASH 0: INK 0
160 BEEP .1,0: BEEP .1,1: BEEP
.1,2: BEEP .1,3: BEEP .2,4
165 REM ***DATA ACHOU/GANHOU***
*
170 DATA 13,2,3,5,7,8,10,15,18,
20,22,25,28,30
180 DATA 5,5,15,20,25,30
190 DATA 10,2,3,5,7,12,13,15,17
,20,25
200 DATA 10,2,5,8,10,15,18,20,2
2,23,25
210 DATA 6,0,1,6,13,21,23
220 DATA 8,0,8,10,11,13,18,23,2
8
230 DATA 8,0,6,8,10,16,18,23,28
240 DATA 9,0,1,3,13,21,26,27,28
,30
250 DATA 7,4,9,12,17,19,27,29
260 DATA 6,2,7,12,17,22,27
270 DATA 4,4,14,22,29
280 DATA 7,7,9,12,17,19,24,29
```

```
 290 POKE 23617,0: INPUT AT 0,0;
"Meta o seu nome. DEPOIS PRIMA
 "ENTER" "; LINE n$: RANDOMIZE
 300 REM ***INICIO***
 310 PLOT 6,98: DRAW 235,0: DRAW
 0,-12: DRAW -235,0: DRAW 0,12
 320 FOR G=20 TO 240 STEP 16
 330 BEEP .05,(G+60)/20: PLOT G,
98: DRAW 0,-12
 340 NEXT G
 350 FOR G=1 TO 29 STEP 2
 360 PRINT AT 10,G;"?"
 370 BEEP .02,29-G: NEXT G
 380 PRINT AT 12,5;"META A RESPO
STA CERTA";TAB 5;"DEPOIS PRIMA "
ENTER""
 390 LET TC=0: LET TE=0: LET TOT
=0: LET N=1: LET A=1
 400 REM ***CRIAR CONTA***
 410 PAUSE 150: GO SUB 5
 420 LET F=INT (RND*3)
 430 LET F$=("*" AND F=0)+("+" A
ND F=1)+("-" AND F=2)
 440 IF F=0 THEN LET PRI=INT (RN
D*12): LET SEG=INT (RND*12): GO
TO 470
 450 LET PRI=INT (RND*50): LET S
EG=INT (RND*50)
 460 IF F=2 AND PRI<SEG THEN GO
TO 450
 470 LET F$=STR$ PRI+F$+STR$ SEG
 480 PRINT AT 15,10;"Pergunta ";
N;" e"; OVER 1;CHR$ 8;"´"
 490 PRINT AT 17,12;F$;" = ?"
 500 POKE 23617,240: POKE 23672,
0: POKE 23673,0
 505 INPUT "QUAL A RESPOSTA ?";
LINE A$: LET T=(PEEK 23672+256*P
EEK 23673)/50
 510 IF CODE A$<48 OR CODE A$>57
 THEN BEEP .8,-20: GO TO 505
 520 PRINT : PRINT INK 2;"RESPON
DEU ";A$
 530 PRINT : PRINT "A RESPOSTA C
ERTA E ";VAL F$
 540 IF STR$ VAL F$<>A$ THEN GO
TO 560
 550 GO TO 670
 560 REM ***RESPOSTA ERRADA***
 570 LET N=N+1: LET TE=TE+T: PRI
NT #0; INK 4;"        RESPOSTA ERR
ADA"
 580 FOR G=10 TO -10 STEP -1
 590 BEEP .01,G: BEEP .008,G-10
 600 NEXT G
 610 IF TOT<1 THEN GO TO 650
 620 PRINT AT 10,(31-2*TOT); INK
 INT (RND*6);"█"
 630 BEEP .1,2*TOT: LET TOT=TOT-
1: BEEP .2,2*TOT: IF TOT=0 THEN
GO TO 790
 650 GO TO 400
 660 REM ***RESPOSTA CORRECTA***
 670 LET TC=TC+T: LET N=N+1: BEE
P .1,2*TOT: LET TOT=TOT+1: PRINT
 AT 10,(31-2*TOT); INK INT (RND*
6);"$": BEEP .1,2*TOT: LET A=A+1
 680 IF TOT=15 THEN GO TO 700
 690 GO TO 400
 700 REM ***VICTORIA***
 710 GO SUB 5
 720 FOR G=0 TO 40-2*(N-A): BEEP
 .01,INT (RND*G): PRINT INK RND*
5;AT 14+RND*7,RND*31;"$": NEXT G
 740 FOR G=0 TO 21: BEEP .02,15-
G: PRINT AT G,0; INK RND*5;"$$$$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$"
 750 NEXT G
 755 REM ***DESPEDIDA***
 760 POKE 23617,252: INPUT "OUTR
A VEZ ?(Sim ou Nao)"; LINE A$: I
F A$="" THEN BEEP .5,-20: GO TO
760
 770 IF A$(1)="S" OR A$(1)="s" T
HEN FOR G=0 TO 21: BEEP .02,2*G:
 RANDOMIZE USR 3582: NEXT G: RUN
 90
 780 IF A$(1)<>"N" AND A$(1)<>"n
" THEN BEEP .5,-20: GO TO 760
 790 LET TE=INT TE: LET TC=INT T
C: LET TT=TE+TC: LET A=A-1: LET
N=N-1: PRINT AT 0,0;: FOR M=10 T
O -9 STEP -1: PRINT ,,: BEEP .02
,M: NEXT M
 800 CLS : PRINT TAB 8; INK 2;"T
entou";TAB 16;"Acertou";TAB 25;"
Falhou"
 805 PRINT AT 2,0;"Numero";TAB 9
;N;TAB 17;A;TAB 26;N-A;AT 4,0;"T
empo";TAB 9;TT;TAB 17;TC;TAB 26;
TE;AT 6,0;" N %";TAB 9;"-";TAB 1
7;INT (A/N*1000)/10;TAB 26;INT (
(N-A)/N*1000)/10;AT 8,0;" T %";T
AB 9;"-";TAB 17;INT (TC/TT*1000)
/10;TAB 26;INT (TE/TT*1000)/10;A
T 10,0;"T / N";TAB 9;INT (TT/N*1
0)/10;TAB 17;INT (TC/A*10)/10;TA
B 26;: IF N<>A THEN PRINT INT (T
E/(N-A)*10)/10
 810 PRINT #1;" ADEUS ";N$: BEEP
 5,45: STOP
9999 POKE 23658,8: POKE 23609,50
: PAPER 6: BORDER 6: INK 0: RUN
90
```

DADOS

SPECTRUM

Programa que simula o JOGO DE DADOS e em que existe valor de aposta, contra o banqueiro SPECTRUM.

```
   0>REM  Programa convertido pa
ra o Spectrum.
   1 GO SUB 7000
   3 BORDER 5: PAPER 5: INK 0: C
LS
   5 PRINT AT 2,8; PAPER 6; INK
1;":";AT 2,10;"JOGO DE DADOS";AT
 2,24;"B"
   8 GO SUB 500
  10 CLS
  15 LET M=30
  20 LET N=M
  30 CLS : LET Z=5
  35 LET P=0
  40 GO SUB 300
  50 GO SUB 400
  60 LET C=A+B
  65 LET j=0
  70 PRINT AT 11+j,0+j;"EU: ";:
GO SUB 6000: PRINT AT 11+j,14+j;
"= "; PAPER 7; INK 2;C
  80 INPUT B$
  81 LET P=P+1
  90 GO SUB 400
 100 LET D=A+B
 105 LET j=5
 110 PRINT AT 11+j,0+3;"VOCE: ";
: GO SUB 6000: PRINT AT 11+j,14+
j;"= "; PAPER 7; INK 2;D
 113 PAUSE 250: CLS
 115 IF P=30 THEN GO TO 421
 120 IF D>C THEN GO TO 160
 130 LET M=M-Z
 140 LET N=N+Z
 150 GO TO 40
 160 LET M=M+Z
 170 LET N=N-Z
 180 GO TO 40
 320 PRINT AT 3,0;"VOCE TEM "; P
APER 6; INK 1;"ESC.:";M;: PRINT
AT 6,15; PAPER 5; INK 0;"EU TENH
O "; PAPER 6; INK 1;"ESC.:";N
 330 PAUSE 60
```

```
 340 RETURN
 400 LET A=1+INT (RND*6)
 405 LET B=1+INT (RND*6)
 410 LET A1=A+1
 415 LET B1=B+1
 420 RETURN
 421 IF M<=N THEN GO TO 428
 422 IF M>N THEN GO SUB 600
 425 GO TO 30
 428 PAUSE 100
 430 CLS
 435 PRINT ,," Terminou o seu te
mpo."''" Quer continuar ? ( s ou
 n )": INPUT l$
 450 IF l$="s" THEN PRINT ''" De
ve ";N-30;" Escudos.": PAUSE 100
 : GO TO 30
 455 IF N>M THEN PRINT ''" Tem a
 pagar "; FLASH 1;N-30; " Escudo
s.": STOP
 460 IF M>N THEN PRINT ''" Tem a
 receber "; FLASH 1;M-30;" Escud
os.": STOP
 500 PRINT AT 5,0;" VOCE vai jog
ar comigo aos Dados"
 510 PRINT ,," Eu, SPECTRUM, sou
  o banqueiro."
 515 PRINT ,," Sempre que haja u
m empate, VOCE"
 520 PRINT ,," Perde. Cada Parad
a e de 5 Escu",," dos."
 525 PRINT ,," B O A  S O R T E
"
 530 PRINT '',," Prima uma Tecl
a."
 535 INPUT L$
 545 RETURN
 600 CLS
 605 PRINT AT 5,1;"VOCE VENCEU.
TEM DIREITO A NOVO"
 610 PRINT AT 8,0;"JOGO, ENTRAND
O COM A VANTAGEM DO"
 615 PRINT AT 11,0;"VALOR JA GAN
HO."
 620 PAUSE 300
 625 CLS
 630 RETURN
6000 LET t$=a$(a1)
6010 LET u$=a$(b1)
6020 PRINT PAPER 6; INK 1;AT 10+
j,4+j;t$(1 TO 3);AT 11+j,4+j;t$(
4 TO 6);AT 12+j,4+j;t$(7 TO 9)
6030 PRINT AT 11+j,8+j;"+ "
6040 PRINT PAPER 6; INK 1;AT 10+
j,10+j;u$(1 TO 3);AT 11+j,10+j;u
$(4 TO 6);AT 12+j,10+j;u$(7 TO 9
)
6200 RETURN
7000 RESTORE 8000
7005 FOR p=0 TO 7
7010 READ a: POKE USR "B"+p,a
7020 NEXT p
7030 DIM a$(7,9)
7040 LET a$(1)="         "
7050 LET a$(2)="    B    "
7060 LET a$(3)="B       B"
7070 LET a$(4)="B   B   B"
7080 LET a$(5)="B B   B B"
7090 LET a$(6)="B B B B B"
7100 LET a$(7)="B BB BB B"
7110 RETURN
8000 DATA 0,0,0,BIN 00011000,BIN
 00011000,0,0,0
9990 STOP
9999 SAVE "DADOS" LINE 1
```

DESENHO TÉCNICO

SPECTRUM

Tradução e adaptação: CLUBE Z 80

Introduza a listagem no computador e faça RUN. A figura definida na linha 9000 DATA, passará por 4 Displays, com a altura que lhe for atribuída logo no início do programa.

1. ROTAÇÃO

A figura move-se 90º, numa série de saltos, antes especificados.

2. DESENHO TRABALHADO

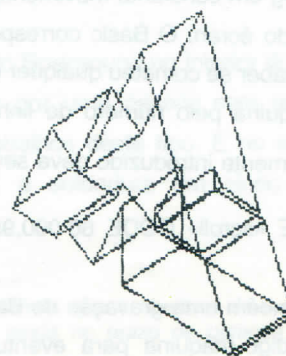
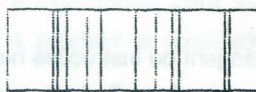
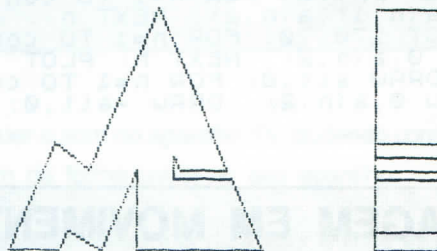
1.º ângulo de projecção ortográfica, de frente, de lado e no plano.

3. DIAGONAL

Display das faces com as linhas a 45º.

4. ISOMÉTRIC

Display de todos os lados com os horizontes em 30º.

Para jogar com o seu desenho dê entrada de uma linha 9000 com as posições tal como numa statement DRAW e finalmente o n.º 1000 que significa o fim de DATA. Separa todos os números por linhas e dá inicialmente as coordenadas do canto inferior esquerdo.

```
  10 BORDER 1: PAPER 1: INK 7: C
LS : RESTORE : GO TO 1000
  40 DIM a(100,2): DIM b(100): L
ET cont=0: LET a1=0: LET a2=0: L
ET a3=0
  50 READ c: IF c=1000 THEN GO T
O 61
  51 LET a1=a1+c: IF a1<a2 THEN
LET a2=a1
  52 IF a1>a3 THEN LET a3=a1
  60 LET cont=cont+1: READ d: LE
T a(cont,1)=c: LET a(cont,2)=d:
GO TO 50
  61 LET comp=a3-a2
  69 REM rotacao
  70 INPUT "Altura, em pixels (2
-50) ?";alt: IF alt<2 OR alt>50
THEN GO TO 70
  72 CLS
  75 PRINT AT 0,0;"Rotacional":
INPUT "Numero de posicoes (2- )?
"'pos: LET pos=INT pos: IF pos<2
 THEN GO TO 75
  76 LET pos=pos-1
  77 LET pp=(127+comp/2)-a3
  80 FOR n=0 TO PI/2+.001 STEP P
I/(2*pos)
  90 FOR m=1 TO cont: LET b(m)=a
(m,2)*COS n: NEXT m
  95 LET a=(INT (((PI/2-n)*(90/P
I*2))*100+.5))/100
 100 CLS : PRINT "Angulo: ";a;"
": PLOT pp,70: FOR m=1 TO cont:
DRAW a(m,1),b(m): NEXT m: IF n=0
 THEN GO TO 200
 150 LET x=alt*SIN n: PLOT pp,70
: DRAW 0,-x: FOR m=1 TO cont: DR
AW a(m,1),b(m): DRAW 0,x: DRAW 0
,-x: NEXT m
 200 INPUT "Copia ? (s/n) "; LIN
E k$: IF k$="s" THEN COPY
 220 NEXT n
 225 REM Desenho
 230 LET pp=ABS a2+10
 250 CLS : PRINT AT 0,0;"TRABALH
AR O DESENHO"
 260 PLOT pp,70: FOR n=1 TO cont
: DRAW a(n,1),a(n,2): NEXT n
 270 PLOT 170,70: FOR n=1 TO con
t: DRAW 0,a(n,2): NEXT n: PLOT 1
70,70: DRAW alt,0: FOR n=1 TO co
nt: DRAW 0,a(n,2): DRAW -alt,0:
DRAW alt,0: NEXT n
 280 PLOT pp,10: FOR n=1 TO cont
: DRAW a(n,1),0: NEXT n: PLOT pp
,10: DRAW 0,alt: FOR n=1 TO cont
: DRAW a(n,1),0: DRAW 0,-alt: DR
AW 0,alt: NEXT n
 290 INPUT "Copia ? (s/n) "; LIN
E k$: IF k$="s" THEN COPY
 300 REM diagonal
 310 LET x=COS (PI/4)*(alt/2): L
ET pp=120+comp/2
 315 CLS : PRINT AT 5,0;"Diagona
l"
 320 PLOT pp,10: FOR n=1 TO cont
: DRAW a(n,1),a(n,2): NEXT n: PL
OT pp,10: DRAW x,x: FOR n=1 TO c
ont: DRAW a(n,1),a(n,2): DRAW -x
,-x: DRAW x,x: NEXT n
 330 INPUT "Copia ? (s/n) "; LIN
E k$: IF K$="s" THEN COPY
 499 REM isometrica
 500 CLS : RESTORE 9000: PRINT "
Isometrica"
 520 DIM c(50,2): LET cont=0
 525 READ c: IF c=1000 THEN GO T
O 600
 530 LET cont=cont+1: READ d
 540 LET c(cont,1)=c*COS (PI/6):
 LET c(cont,2)=d+(-c*SIN (PI/6))
 550 GO TO 525
 610 PLOT 150,0
 620 FOR n=1 TO cont: DRAW c(n,1
),c(n,2): NEXT n
 630 LET e=alt*COS (PI/6): LET f
=alt*SIN (PI/6)
 640 PLOT 150,0: DRAW e,f
 650 FOR n=1 TO cont: DRAW c(n,1
);c(n,2): DRAW -e,-f: DRAW e,f:
NEXT n
 660 INPUT "Copia ? (s/n) "; LIN
E k$: IF k$="s" THEN COPY
 999 STOP
1000 FOR n=0 TO 7: READ a: POKE
USR "a"+n,a: NEXT n: DATA 48,72,
72,48,0,0,0,0
1050 PRINT TAB 8;"DESENHO TECNIC
O": PAUSE 50: GO TO 40
9000 DATA -55,0,0,30,-15,-30,-15
,7,-4,-7,-22,0,20,40,15,-10,30,6
0,30,-60,-20,0,-3,5,0,-8,25,0,14
,-27,1000
```

MENSAGEM EM MOVIMENTO

SPECTRUM 16/48

Trad. e adapt./CLUBE Z80

Já pensou em obter uma mensagem ou instruções num jogo ou num utilitário na 23 linha do écran? Aqui tem essa oportunidade.

O programa em código-máquina (list. 1) pode ser usado para apresentar um string em constante movimento (scroll) de um lado para o outro do écran. O Basic correspondente a esta rotina permite-lhe saber se cometeu qualquer erro na introdução do código-máquina pelo número de linha.

Assim que correctamente introduzido deve ser gravado com:

SAVE «scroll» CODE 60 000,95

É recomendada também uma gravação do Basic, de forma a ter acesso ao código-máquina para eventuais alterações.

***

Para iniciar a introdução da listagem 2 pode usar o comando NEW, já que o código-máquina estará presente e posicionado acima da RAMTOP.

Terminada esta 2.ª operação faça: RUN.

(Deverá obter o scroll da 23.ª linha do conteúdo de Z$, pixel/pixel).

Para regressar ao BASIC, accione qualquer tecla.

***

Para incorporar esta rotina nos seus programas, pode usar a subrotina (2.ª listagem), definindo a sua mensagem em Z$, utilizando o comando GO SUB.

O código-máquina é totalmente recolocável, mas não deve ser carregado em endereços já ocupados.

Se pretender o scroll da mensagem uma só vez, altere o valor 40 na linha 70 da 1.ª listagem para 200.

Se possui o Spectrum 16 K, adicione as linhas seguintes ao segundo programa Basic.

15 RANDOMIZE (novo endereço)

16 POKE 60001, PEEK 23670 : POKE 60002, PEEK 23671

Deve alterar também o valor 59399 nas linhas 30 e 50 para o novo valor (novo endereço).

NOTE: Quando Z$ é colocado em memória (POKE), é importante que o último endereço seja «POKED» com 0. Desta forma a rotina saberá onde é o fim da mensagem. Repara na linha 50 da 2.ª listagem.

Listagem 1

```
  10 DATA 33,7,232,34,0,91,24,40
,62,8,531
  20 DATA 245,6,6,197,6,255,16,2
54,193,16,1194
  30 DATA 248,33,255,88,229,14,8
,6,32,225,1138
  40 DATA 37,229,183,203,22,43,1
6,251,13,32,1029
  50 DATA 242,225,241,61,32,220,
24,214,255,58,1572
  60 DATA 5,92,254,0,192,42,0,91
,35,34,745
  70 DATA 0,91,126,254,0,40,189,
33,0,60,793
  80 DATA 1,8,0,9,61,32,252,17,2
55,80,715
  90 DATA 237,160,27,20,121,254,
0,32,247,24,173,1295
  95 CLEAR 59399
 100 LET b=0
 110 FOR n=60000 TO 60091
 120 READ z
 130 IF z>255 THEN LET d=b: LET
b=0: IF z<>d THEN PRINT "ERRO NA
 LINHA ";(INT ((N-60000)/10))*10
 140 IF n=60091 THEN STOP
 150 IF z>255 THEN GO TO 120
 160 POKE n,z
 170 LET b=b+z
 180 NEXT n
```

Listagem 2

```
  10 LET z$="QUALQUER TECLA PARA
 PARAR ESTA DEMONSTRACAO. PARA S
ABER O COD. DA TECLA USADA - PEE
K 23560"
  20 FOR n=1 TO LEN z$
  30 POKE 59399+n,CODE z$(n)
  40 NEXT n
  50 POKE n+59399,0
  60 RANDOMIZE USR 60000
  70 PRINT "ESSA TECLA TEM O COD
 ";PEEK 23560;
```

SPECSOUND

Esc. 1696$50

COMO OBTER O SOM DO SPECTRUM DIRECTAMENTE NO SEU TV, ATRAVÉS DO CABO DA ANTENA.

1. GERAL

O ZX-Spectrum possui notáveis qualidades gráficas, o que lhe permite executar programas de jogos, e outros, com grande realismo. No entanto, as também boas qualidades de programação de som não são devidamente aproveitadas. A saída de som faz-se por um pequeno altifalante interior, dificilmente audível.

O módulo Specsound, montado no interior do seu Spectrum, destina-se a resolver este problema, modulando a sub-portadora de som do canal TV do Spectrum com a informação sonora produzida pelo programa. Assim terá quer a imagem quer o som no aparelho TV, podendo controlar o volume de som da forma usual no seu aparelho.

Os acessórios externos, para ligar à ficha traseira do Spectrum, que também se encontram no mercado, são mais complexos e incómodos, para além de mais caros, nunca chegando a produzir a qualidade sonora desejável.

3. MÉTODO DE INSTALAÇÃO

A instalação do Specsound no interior do seu Spectrum deve ser efectuada por uma pessoa com um mínimo de experiência em trabalhos deste tipo. É no entanto simples, pois apenas inclui a soldadura em cinco pontos do circuito impresso.

NOTA: Recordamos que a abertura do seu Spectrum, caso este esteja ainda no prazo de garantia, tornará esta inválida.

FIGURA 1

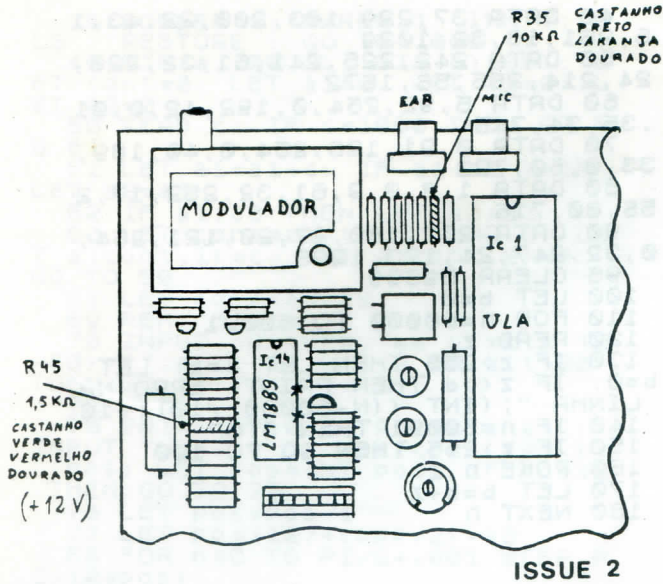
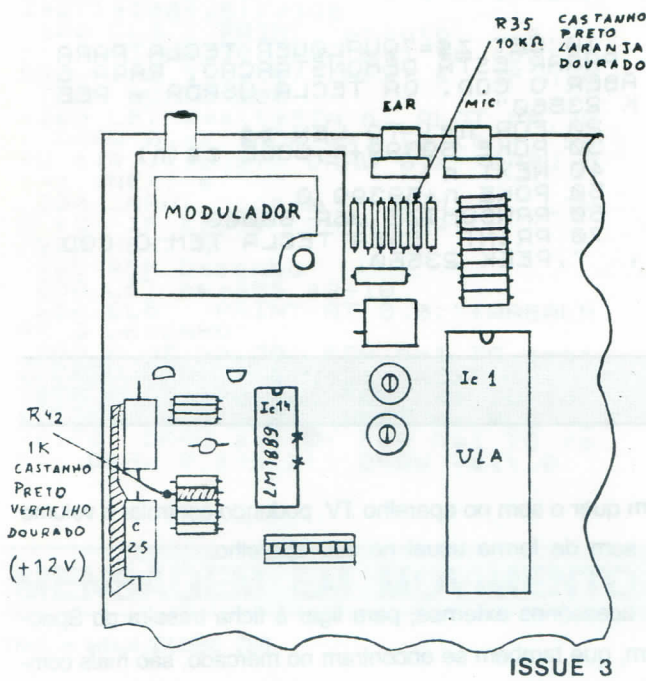
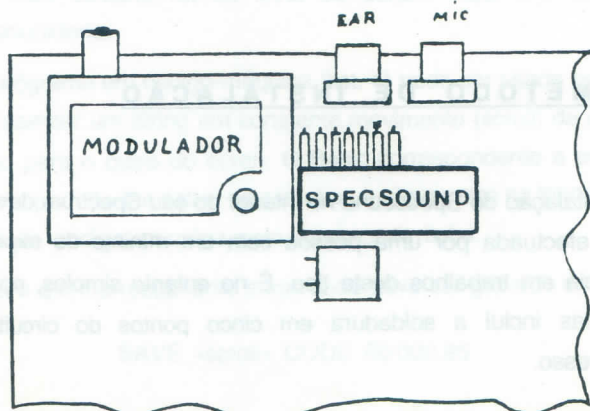
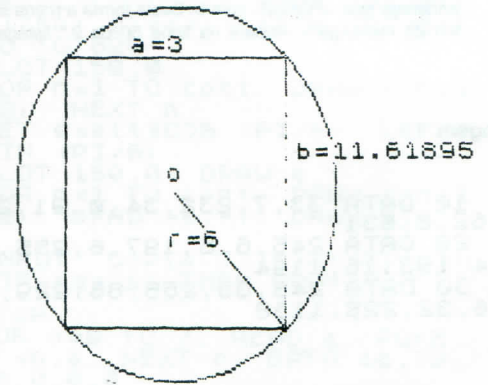
FIGURA 3

DESAFIO

PROGRAMAÇÃO-BASIC

Autor: MANUEL QUINAZ
Porto

Correspondendo à sugestão do Hugo Assumpção, recebemos uma resposta ao desafio proposto — criar um programa que permita calcular a área de um paralelograma rectângulo inserido numa circunferência, dados unicamente o raio desta e um dos lados do paralelograma.

A listagem do programa que publicamos a seguir é da autoria de Manuel Quinaz/Porto.

```
r=Raio da circunferencia
a=Lado do paralelogramo
b=lado do paralelogramo
```

```
  10 CIRCLE 100,100,70
  20 PLOT 52,150: DRAW 96,0
  30 DRAW 0,-100: DRAW -96,0: DR
AW 0,100
  40 PLOT 100,100: DRAW 48,-50
  50 PRINT AT 8,12;"o": PRINT AT
 11,12;"r=?";CHR$ 8;: INPUT "r="
;r: PRINT r
  60 PRINT AT 2,10;"a=?";CHR$ 8;
: INPUT "a=";a: PRINT a
  65 IF 4*(r↑2)<a↑2 THEN INPUT "
Erro nos dados.Contas indetermi-
nadas [Prima ENTER]"; LINE a$: R
UN
  66 IF 4*(r↑2)=a↑2 THEN INPUT "
Nao ha paralelogramo inscrito
na circunferencia [Prima ENTER]"
; LINE a$: RUN
  70 LET b=SQR (4*(r↑2)-a↑2)
  80 PRINT AT 7,19;"b=";b
  90 PRINT AT 19,0;"r=Raio da ci
rcunferencia"'"a=Lado do paralel
ogramo"'"b=lado do paralelogramo
"
 120 PRINT #0;AT 0,0;"o=centro d
a circunferencia"'"Area=b*a=";b*
a
 130 PAUSE 1: PAUSE 0: RUN
```

NOVOS PROGRAMAS SPECTRUM

- SHERLOCK HOLMES — Trata-se de uma das melhores aventuras para o Spectrum, que apareceu após a inesquecível «The Hobbit». Desta vez, a personagem celebérrima de Sherlock Holmes tem que resolver um dos seus mais movimentados mistérios policiais. As instruções para a execução deste programa são as normais para qualquer tipo de aventura do género; com a diferença de que esta aventura aceita cerca de 1500 vocábulos !!!

- B. C. BILL — B. C. BILL é um jogo que se passa na idade da pedra. Você é BILL, um homem das cavernas e tem que se defender dos inimigos para arranjar comida e mulheres. Deverá para isso andar à cautela e levar as presas para a sua caverna. B. C. BILL tem além de tudo a qualidade de um jogo do Imagine Software sendo dos melhores do género. Quando acabar de fazer o LOAD deverá carregar em qualquer tecla para escolher entre Joysticks ou teclado. Conforme a sua opção carregue no número correspondente e para confirmar accione Y.

  TECLAS: Q — subir; A — descer; 1 — tacada; M — esquerda; N — direita.

- AVALON — É um jogo que combina a aventura com a acção, de uma forma que precisa de pensar, de desenhar um mapa e ao mesmo tempo agir, pois os inimigos vão-lhe roubando energia. Vai encontrar cerca de 200 salas, túneis e caves em 8 níveis, que deve explorar para derrotar o senhor do CAOS. Aparecem objectos e feitiços que o poderão ajudar. Algumas portas não se abrem sem chave, outras permanecem invisíveis até resolver um determinado problema. Na parte inferior do écran pode ver a qualquer momento os objectos e feitiços que possui. O programa pode ser jogado com Joysticks.

- TIME BOMB — A nossa missão é desactivar as bombas relógio, existentes no écran, antes que o tempo chegue ao fim. Se a bomba relógio não for desactivada a tempo, ela explodirá provocando uma morte lenta e dolorosa. Não se pode passar duas vezes pelo mesmo sítio e tem que se evitar os SKULLS (pr.nível) e as botas (níveis mais avançados). Podemos selecionar os níveis de dificuldade, bastando para isso e antes do jogo começar, carregar na tecla de 1 a 5, consoante se quiser um nível fácil ou difícil, respectivamente.

  TECLAS: Q — subir; Z — descer; I — Esquerda; P — direita.

- SPACE COMMAND — Neste jogo, temos que defender uma base contra diversos inimigos e perigos; naves, meteoritos, alienigenes, etc. . . . Temos também que evitar que a nossa nave choque com esses objectos e para os destruir, devemos disparar o laser de que dispomos a bordo da nave. Há diversos níveis de dificuldade e uma variedade considerável de écrans. O programa é compatível com Joysticks ou teclado. Através da tecla G o programa fornece a demonstração de vários níveis.

  TECLAS: SPACE — fogo; CAPS — descer; M — esquerda; S — Recomeçar; Z — subir; SYMBOL — direita.

- JUMP CHALLENGE — Neste jogo você tem que dar saltos, com uma bicicleta e depois com uma moto por cima de carros e obstáculos. Para se poder classificar tem que saltar primeiro com uma bicicleta, podendo escolher a distância que terá que percorrer até à barreira. Para isso basta carregar em 'CAPS SHIFT', que serve para acelerar, e depois de deixar que a bicicleta que deve carregar em 'A', que é o sinal que tem que dar ao computador para que se comece o salto. Depois de 'A' devemos carregar outra vez em 'CAPS' para acelerar e saltar. A bicicleta para saltar todos os obstáculos, deve ir no máximo de velocidade. Depois vem a vez da moto. O procedimento para o caso da moto é análogo, tendo apenas mais teclas para o funcionamento: a tecla 'Z' que actua como sendo o travão, a tecla 'SIMBOL SHIFT' que serve para deslizar um 'cavalo' (salto da mota e o seu andamento apenas sobre a roda traseira) e a tecla 'SPACE' que serve para fazer os cavalos. Atenção, você só tem uma vida, por isso evite cair com a mota. BOA SORTE.

  TECLAS: A — Preparar para o salto; Z — Travar; CAPS — Acelarar; SYMBOL — Desfazer o 'cavalo'; SPACE — Fazer 'cavalo'.

- DANGER MOUSE — Neste jogo, que é uma reprodução de uma série de desenhos animados americanos, você vai encarnar a personagem de DANGER MOUSE, um rato que se põe ao serviço do bem-estar da humanidade.
  Para isso, ele possui vários apetrechos, que vocês podem utilizar, desde que as utilizem tão bem como o próprio DANGER MOUSE. O jogo tem dois níveis de dificuldade: o fácil, para principiantes, e o profissional, para DANGER MOUSES experimentados. O jogo tem também três quadros diferentes. No primeiro, temos que eliminar os inimigos que vêem contra nós, bastando para isso pôr-nos à frente deles. No 2.° temos que atravessar um lago com um crocodilo saltando por cima do mesmo e passando à continuação do 2.° quadro em que temos que escapar às terríveis aranhas gigantes e a um enorme cão. No 3.° quadro temos que apagar as lâmpadas que se vão acendendo à nossa frente, bastando para isso saltar para as que estão imediatamente à nossa frente e mais próximas. Se passarmos os três quadros, teremos salvo o mundo, e voltamos ao início do jogo, para mais uma aventura.

  TECLAS: *(Nível 1):* 8 — subir; 9 — descer. *(Nível 2 e 3):* 0 — saltar. 1 — esquerda; 2 — direita.

NOTA: Este jogo é especialmente aconselhado para crianças.

- MONTY MOLE — Neste jogo, você tem que descobrir os pedaços de carvão de pedra necessários para conseguir destruir e eliminar todos os perigos, armadilhas, seres e objectos que for encontrando e, no final, conseguir vencer os seres que dominam a ponte (logo no 1.° quadro), reparar a ponte e atear fogo ao caldeirão do outro lado da margem. Cuidado com as armadilhas existentes na passagem de um quadro para o outro. Evite os objectos que encontrar (excepto o carvão e outros que lhe darão energia), tais como spray, prensas, etc. . . . Para subir nas cordas, ou em patamares basta carregar na tecla para saltar.

  TECLAS: Q — subir; O — esquerda; M — saltar; A — descer; P — direita.

- DECATHLON - DIA 2 — Trata-se de um programa que simula algumas das provas mais interessantes do Atletismo. Consiste na continuação do Dia 1 (Decathlon) com as provas de 110 m barreiras, salto em altura, lançamento do disco, lançamento do dardo, etc..

  COMANDOS: > H — pe esq.; M — pe dir.; Symb. Shift — saltar e lançar.

- POLE POSITION — Este programa é uma réplica da famosa máquina de «ATARI». O jogo consiste em conduzir um carro de fórmula 1 ao longo de um circuito, sendo a 1.ª volta para classificação e distribuição dos oito lugares na grelha de partida. Depois da 1.ª volta (de classificação) e de serem distribuidos os lugares, temos que percorrer o circuito mais 4 vezes, finda as quais nós poderemos, ou não, entrar para a tabela de pontuações máxima. Sempre que completarmos uma volta, recebemos um bónus em tempo. O jogo só acaba se se acabar o tempo.

  TECLAS: P — Direita; A — Velocidades: S — Começar; O — Esquerda; Q — Travar.

- KOKOTONI WILF — Jogo de aventuras, no qual temos que guiar o KOKOTONI, um lendário homem com asas, através de vários cenários, evitando todo o contacto com outros seres e animais pré-históricos, ao mesmo tempo que se vai recolhendo item nos vários quadros. O máximo de item (1 por quadro) e guiar o KOKOTONI para o centro da terra, onde ele habita.

  TECLAS: 6 — esquerda; 1 — voar; 7 — direita.

- DRILLER TANKS — O objectivo deste jogo e o de limpar o túnel, bastando para isso passar com a escavadora por ele. Temos que evitar os MAMMUTS e os SKORKS, e que eles atinjam o palácio. Podemos destruir tanto os MAMMUTS como os SKORKS, desintegrando-os. Cuidado, se os MAMMUTS atingirem o palácio, seremos totalmente destruidos e perdemos o jogo. Se um SKORK se atravessar à nossa frente, teremos o caminho obstruido. O jogo e compatível tanto com o Joystick como com o teclado.

- MACHINE CODE TUTOR (800$00) — Este programa, procura numa série de ligações progressivas, explicar e exemplificar a aplicação das instruções referentes ao microprocessador Z80, com a particularidade de permitir entrar nos exemplos, sem ocorrer o crash do sistema. As lições são apresentadas em 4 blocos que deverão ser carregados cada um por sua vez, usando para isso a última opção de cada MENU. Assim:

1.ª Lição (1-9)

1. Registos e memória
2. Simples instrução de carregamento (LOAD)
3. Par de registos
4. Endereço indirecto
5. A adição e o carrey flag
6. A subtracção e o carry flag
7. Incremento e decremento
8. Flag 0
9. Comparação

2.ª Lição (10-17)

Nesta lição, pretende-se ilustrar a diferença entre o computador e a calculadora.

10. Saltos condicionais e incondicionais
11. Saltos relativos
12. O stack
13. Chamada de subrotinas
14. Notação binária
15. Notação hexadecimal
16. Código binário — notação decimal
17. Notação positiva e negativa

3.ª Lição (18-25)

Este grupo trata da manipulação do bit e a sua aplicação nas operações flag e nos rápidos métodos de multiplicação.

18. Paridade
19. O registo Flag e AF
20. Flags S e P/V em instrução
21. Manipulação do bit
22. Instruções lógicas
23. Instruções shift
24. Instruções rotativas
25. Rotação decimal

4.ª Lição (26-35)

Neste bloco são apresentados mais registos e instruções e como o Z80 contacta com o exterior.

26. Indice de registos
27. Colocação dos registos
28. Instrução INPUT/OUT PUT
29. Bloco de instruções
30. Bloco de transferência de instruções
31. Bloco de procura de instruções
32. Bloco INPUTT/OUT PUT de instruções
33. Processador de controlo de instruções
34. Interrupção
35. Final

PERGUNTAS/OBSERVAÇÕES/COMENTÁRIOS

JOÃO MARQUES / Barreiro

Faço um reparo em relação aos termos usados na vossa circular, em que alertam para o fim da assinatura, concretamente o 1.º parágrafo.

CLUBE Z80

Nem sempre quem escreve uma circular usa os termos mais simpáticos ou diplomáticos. Nunca nos passou pela mente chamar a atenção dos assinantes do jornal para o facto do fim da assinatura ter chegado, em termos que não fossem muito convenientes. De qualquer modo as nossas deculpas.

JOÃO MARQUES / Barreiro — SUGESTÃO

Não será possível irem publicando os indices dos números atrasados da revista? Isso iria permitir aos sócios mais recentes terem uma ideia de quais os números que eventualmente lhes podem interessar.

CLUBE Z 80

Faremos os possíveis para corresponder à sua sugestão.

ARMANDO BEÇA / Bragança

Ando a fazer um jogo que necessita de saber se foram primidas várias teclas ao mesmo tempo e quais foram essas teclas. Alguém me poderá indicar um processo de conseguir esse efeito usando a linguagem máquina?

RS 232/INTERFACE PARA COMUNICAÇÕES

Temos sido regularmente solicitados a explicar o que é o interface RS 232 e qual o seu interesse. Espanta-nos por vezes que inclusive, pessoas ligadas à comercialização de equipamentos e microcomputadores não tenham o conhecimento simples e imediato para explicar ao utilizador e cliente o interesse em possuir esse tipo de interface. Para ajudar uns e outros, segue-se a explicação.

O circuito electrónico RS 232 foi originalmente desenhado para permitir aos computadores, receber ou enviar dados, através da linha telefónica e usando um MODEM.

Todavia, actualmente observa-se com muita frequência, a ligação directa entre dois ou mais computadores, e entre estes e diversos tipos periféricos como impressoras, traçadores de gráficos (plotters), etc., tendo como tradutor destas ligações o interface RS 232.

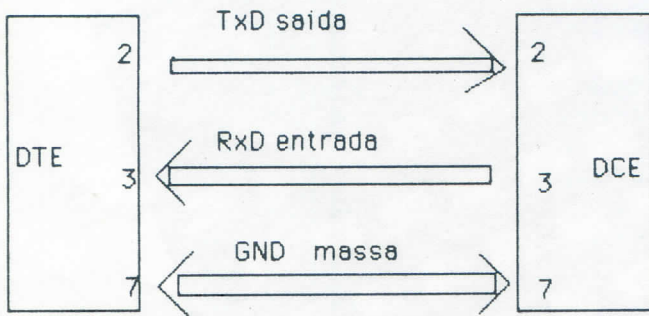
O chamado standard RS 232, manifesta-se de diversas formas, de acordo com o projectista ou fabricante e pode tornar-se numa tarefa difícil e tediosa estabelecer pela primeira vez a comunicação entre dois equipamentos diferentes, embora ambos possuindo ou sendo equipados com RS 232.

Veremos então (e recorrendo aos estudos do Eng. António Fonseca) os problemas básicos, encontrados na tentativa de estabelecer esse tipo de comunicação.

O standard ou padrão RS 232-C refere-se a dois tipos de equipamento:

1. Data Terminal Equipment (DTE)
2. Data Comunications Equipment (DCE).

À partida temos de admitir que quer o terminal (habitualmente o DTE) e o MODEM (normalmente o DCE) possuem o mesmo tipo de ficha DIN 25 pinos ou DIN 9 pinos.

MODEM... anagrama de MOdulador DEmodulador = modulador - demodulador de circuitos telefónicos separados, cada um dos quais contém um ou mais canais telefónicos.

A figura acima mostra como o DTE transmite os dados desde o pino 2, enquanto o DCE recebe os mesmos também no pino 2 — por isso essa linha recebe o nome de TxD ou seja Tranmit Data ou transmissão de dados.

Entretanto, no pino 3 são recebidos dados no DTE e o dispositivo DCE transmite esses mesmos dados através do pino 3 — por isso essa linha chama-se RxD ou Receive Data ou recepção de dados.

Isto torna-se de certo modo confuso, e origina uma encrenca de todo o tamanho quando os pinos e os dispositivos estão classificados de modo que não sabemos designar qual o DTE e qual o DCE.

Infelizmente, alguns fabricantes insistem em designar os seus computadores como dispositivos do tipo DCE enquanto na documentação de outros, a classificação é de dispositivos tipo DTE. Isto obviamente dificulta a ligação das portas série em cada equipamento.

Se uma porta (ficha de saída/entrada) estiver configurada como DCE terá as designações que chamamos SER1, enquanto que se estiver designada como DTE será a lista agrupada como SER2 que interessa anotar.

SER 1			SER 2		
PINO	**NOME**	**FUNÇÃO**	**PINO**	**NOME**	**FUNÇÃO**
1	GND	massa ou terra	1	GND	massa ou terra
2	TxD	entrada	2	TxD	saída
3	RxD	saída	3	RxD	entrada
4	DTR	pronto p/ entrada	4	DTR	pronto p/ saída
5	CTS	pronto p/ saída	5	CTS	pronto p/ entrada
6		+ 12 volt	6		+ 12 volt

nota: DTR = terminal de dados pronto
CLS = limpar para enviar

TOP 10 EM INGLATERRA

(«YOUR SPECTRUM» 10 DEZ/JAN 85)

1. JET SET WILLY
2. SABRE WULF
3. MANIC MINER
4. ATIC ATAC
5. TRASHMAN
6. ANT ATTACK
7. CHEQUERED FLAG
8. LUNAR JETMAN
9. WHEELIE
10. LORDS OF MIDNIGHT

NO CLUBE Z80

(MAIS VENDIDOS)

1. BEACH-HEAD
2. DECATHLON (1 e 2)
3. FULL TROTTLE
4. PHEENIX
5. MATCH POINT
6. POLE POSITION
7. PSYTRON
8. SABRE WULF
9. TLL
10. GLUG GLUG